DB2 Universal Database for z/OS

Version 8

Command Reference

Information Management software

SC18-7416-05

DB2 Universal Database for z/OS

Version 8

Command Reference

Information Management software

SC18-7416-05

Note
Before using this information and the product it supports, be sure to read the general information under "Notices" on page 447.

Sixth Edition, Softcopy Only (February 2008)

This edition applies to Version 8 of IBM DB2 Universal Database for z/OS (DB2 UDB for z/OS), product number 5625-DB2, and to any subsequent releases until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

This softcopy version is based on the printed edition of the book and includes the changes indicated in the printed version by vertical bars. Additional changes made to this softcopy version of the book since the hardcopy book was published are indicated by the hash (#) symbol in the left-hand margin. Editorial changes that have no technical significance are not noted.

This and other books in the DB2 UDB for z/OS library are periodically updated with technical changes. These updates are made available to licensees of the product on CD-ROM and on the Web (currently at www.ibm.com/software/data/db2/zos/library.html). Check these resources to ensure that you are using the most current information.

© Copyright International Business Machines Corporation 1983, 2008. All rights reserved.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this book vii

Summary of changes to this book xv

Part 1. Privileges, authorization IDs, and the bind process 1

Chapter 1. Privileges and authorization IDs 3

Chapter 2. The bind process 5

Part 2. Working with commands 7

Chapter 3. DB2 command parsing 9

Chapter 4. Scope of commands 13

Chapter 5. Output from DB2 commands 15

Chapter 6. Issuing commands to DB2 from IFI. 17

Chapter 7. DSN subcommand parsing 19

Chapter 8. Description of commands. 21

Part 3. Commands 27

Chapter 9. -ALTER BUFFERPOOL (DB2) 29

Chapter 10. -ALTER GROUPBUFFERPOOL (DB2) 35

Chapter 11. -ALTER UTILITY (DB2) 39

Chapter 12. -ARCHIVE LOG (DB2) 43

Chapter 13. BIND PACKAGE (DSN). 49

Chapter 14. BIND PLAN (DSN). 57

Chapter 15. BIND and REBIND options 63

Chapter 16. -CANCEL THREAD (DB2) 97

Chapter 17. /CHANGE (IMS) 103

Chapter 18. DCLGEN (DECLARATIONS GENERATOR) (DSN) 105

Chapter 19. /DISPLAY (IMS) 115

© Copyright IBM Corp. 1983, 2008

Chapter 20. -DISPLAY ARCHIVE (DB2) 119
Chapter 21. -DISPLAY BUFFERPOOL (DB2) 121
Chapter 22. -DISPLAY DATABASE (DB2) 131
Chapter 23. -DISPLAY DDF (DB2) 149
Chapter 24. -DISPLAY FUNCTION SPECIFIC (DB2) 153
Chapter 25. -DISPLAY GROUP (DB2) 157
Chapter 26. -DISPLAY GROUPBUFFERPOOL (DB2) 163
Chapter 27. -DISPLAY LOCATION (DB2) 177
Chapter 28. -DISPLAY LOG (DB2) 181
Chapter 29. -DISPLAY PROCEDURE (DB2) 183
Chapter 30. -DISPLAY RLIMIT (DB2) 189
Chapter 31. -DISPLAY THREAD (DB2) 191
Chapter 32. -DISPLAY TRACE (DB2) 205
Chapter 33. -DISPLAY UTILITY (DB2) 211
Chapter 34. DSN (TSO) 217
Chapter 35. DSNC (CICS attachment facility) 221
Chapter 36. DSNC DISCONNECT (CICS attachment facility) 223
Chapter 37. DSNC DISPLAY (CICS attachment facility) 225
Chapter 38. DSNC MODIFY (CICS attachment facility) 229
Chapter 39. DSNC STOP (CICS attachment facility) 231
Chapter 40. DSNC STRT (CICS attachment facility) 233
Chapter 41. DSNH (TSO CLIST) 235
Chapter 42. END (DSN) 265
Chapter 43. FREE PACKAGE (DSN) 267
Chapter 44. FREE PLAN (DSN) 271
Chapter 45. MODIFY admtproc,APPL=SHUTDOWN (z/OS) 273
Chapter 46. MODIFY admtproc,APPL=TRACE (z/OS) 275

Chapter 47. MODIFY irlmproc,ABEND (z/OS IRLM) 277
Chapter 48. MODIFY irlmproc,DIAG (z/OS IRLM) 279
Chapter 49. MODIFY irlmproc,PURGE (z/OS IRLM) 281
Chapter 50. MODIFY irlmproc,SET (z/OS IRLM) 283
Chapter 51. MODIFY irlmproc,STATUS (z/OS IRLM) 287
Chapter 52. -MODIFY TRACE (DB2) 295
Chapter 53. REBIND PACKAGE (DSN) 299
Chapter 54. REBIND PLAN (DSN) 303
Chapter 55. REBIND TRIGGER PACKAGE (DSN) 307
Chapter 56. -RECOVER BSDS (DB2). 311
Chapter 57. -RECOVER INDOUBT (DB2) 313
Chapter 58. -RECOVER POSTPONED (DB2) 317
Chapter 59. -RESET GENERICLU (DB2) 321
Chapter 60. -RESET INDOUBT (DB2) 323
Chapter 61. RUN (DSN) 327
Chapter 62. -SET ARCHIVE (DB2) 331
Chapter 63. -SET LOG (DB2) 335
Chapter 64. -SET SYSPARM (DB2) 339
Chapter 65. SPUFI (DSN) 341
Chapter 66. /SSR (IMS) 343
Chapter 67. START admtproc (z/OS) 345
Chapter 68. STOP admtproc (z/OS) 347
Chapter 69. /START (IMS) 349
Chapter 70. -START DATABASE (DB2) 351
Chapter 71. -START DB2 (DB2) 361
Chapter 72. -START DDF (DB2) 365
Chapter 73. -START FUNCTION SPECIFIC (DB2) 367

Chapter 74. START irlmproc (z/OS IRLM) 371
Chapter 75. -START PROCEDURE (DB2) 377
Chapter 76. -START RLIMIT (DB2). 381
Chapter 77. -START TRACE (DB2) 383
Chapter 78. /STOP (IMS) 395
Chapter 79. -STOP DATABASE (DB2) 397
Chapter 80. -STOP DB2 (DB2) 403
Chapter 81. -STOP DDF (DB2) 405
Chapter 82. -STOP FUNCTION SPECIFIC (DB2) 409
Chapter 83. STOP irlmproc (z/OS IRLM) 413
Chapter 84. -STOP PROCEDURE (DB2) 415
Chapter 85. -STOP RLIMIT (DB2) 419
Chapter 86. -STOP TRACE (DB2) 421
Chapter 87. -TERM UTILITY (DB2). 427
Chapter 88. /TRACE (IMS) 431
Chapter 89. TRACE CT (z/OS IRLM) 433

Part 4. Appendixes 437

Appendix. Directory of subsystem parameters 439
How to use the DB2 library 445
Notices 447
Glossary 451
Bibliography 485
Index 493

About this book

This is a reference book that lists numerous commands that system administrators, database administrators, and application programmers use. The commands are in alphabetical order for quick retrieval.

Unless it is stated otherwise, this information assumes that DB2® is running in new-function mode (as opposed to compatibility mode or enabling-new-function mode).

Important

In this version of DB2 UDB for z/OS, the DB2 Utilities Suite is available as an optional product. You must separately order and purchase a license to such utilities, and discussion of those utility functions in this publication is not intended to otherwise imply that you have a license to them. See Part 1 of *DB2 Utility Guide and Reference* for packaging details.

The DB2 Utilities Suite is designed to work with the DFSORT program, which you are licensed to use in support of the DB2 utilities even if you do not otherwise license DFSORT for general use. If your primary sort product is not DFSORT, consider the following informational APARs mandatory reading:

- II14047/II14213: USE OF DFSORT BY DB2 UTILITIES
- II13495: HOW DFSORT TAKES ADVANTAGE OF 64-BIT REAL ARCHITECTURE

These informational APARs are periodically updated.

Who should read this book

This book presents reference information for people involved in system administration, database administration, and operation. It presents detailed information on commands, including syntax, option descriptions, and examples for each command.

Conventions and terminology used in this book

Naming conventions that are unique to DB2 commands are discussed in "Naming conventions." Terminology is discussed in "Terminology and citations" on page x.

Naming conventions

When a parameter refers to an object created by SQL statements (for example, tables, table spaces, and indexes), SQL syntactical naming conventions are followed.

This section describes naming conventions unique to commands. Characters are classified as *letters*, *digits*, or *special characters*.

- A *letter* is any one of the uppercase characters A through Z (plus the three characters reserved as alphabetic extenders for national languages, #, @, and $ in the United States).
- A *digit* is any one of the characters 0 through 9.

© Copyright IBM Corp. 1983, 2008

- A *special character* is any character other than a letter or a digit.

See Chapter 2 of *DB2 SQL Reference* for an additional explanation of identifiers.

authorization-id
An identifier of 1 to 128 letters, digits, or the underscore that identifies a set of privileges. An authorization ID must begin with a letter.

collection-id
An SQL identifier of 1 to 128 letters, digits, or the underscore that identifies a collection of packages; therefore, a collection ID is a qualifier for a package ID. A collection ID must begin with a letter.

A collection ID should not begin with DSN; this can sometimes conflict with DB2-provided collection IDs. If a collection ID beginning with DSN is specified, DB2 issues a warning message.

connection-name
An identifier of one to eight characters that identifies an address space connection to DB2. A connection identifier is one of the following:

- For DSN processes running in TSO foreground, the connection name "TSO" is used.
- For DSN processes running in TSO batch, the connection name BATCH is used.
- For the call attachment facility (CAF), the connection name DB2CALL is used.
- For the Resource Recovery Services attachment facility (RRSAF), the connection name RRSAF is used.
- For IMS and CICS processes, the connection name is the system identification name.

See Part 4 (Volume 1) of *DB2 Administration Guide* for more information about connection names.

correlation-id
An identifier of 1 to 12 characters that identifies a process within an address space connection. A correlation ID must begin with a letter.

A correlation ID can be one of the following:

- For DSN processes running in TSO foreground, the correlation ID is the TSO logon identifier.
- For DSN processes running in TSO batch, the correlation ID is the job name.
- For CAF processes, the correlation ID is the TSO logon identifier.
- For RRSAF processes, the correlation ID is the value specified during signon.
- For IMS processes, the correlation ID is *pst#.psbname*.
- For CICS processes, the correlation ID is *identifier.thread_number.transaction_identifier.*

See Part 4 (Volume 1) of *DB2 Administration Guide* for more information about correlation IDs.

data-set-name
An identifier of 1 to 44 characters that identifies a data set.

dbrm-member-name
An identifier of one to eight letters or digits that identifies a member of a partitioned data set. (MVS™ requires this naming convention.)

A DBRM member name should not begin with DSN; this can sometimes conflict with DB2-provided DBRM member names. If a DBRM member name beginning with DSN is specified, DB2 issues a warning message.

dbrm-pds-name
An identifier of 1 to 44 characters that identifies a partitioned data set.

ddname
An identifier of one to eight characters that designates the name of a DD statement.

hexadecimal-constant
A sequence of digits or any of the letters from A to F (uppercase or lowercase).

hexadecimal-string
An X followed by a sequence of characters that begins and ends with an apostrophe. The characters between the string delimiters must be a hexadecimal number.

ip address (or Internet address)
A 4-byte value that uniquely identifies a TCP/IP host within the TCP/IP network. IP addresses are usually displayed in a format called *dotted decimal*, where each byte of the IP address is displayed in decimal format with a period delimiting each number.

location-name
A location identifier of 1 to 16 letters (but excluding the alphabetic extenders), digits or the underscore that identifies an instance of a database management system. A location name must begin with a letter.

luname
An SQL identifier of one to eight characters that identifies a logical unit name. An *luname* must begin with a letter.

luwid
A fully qualified LU network name and an LUW instance number.

The LU network name consists of an optional eight character network ID, a period, and an eight character network LU name. If you indicate no network ID, no period is required. The LUW instance number consists of 12 hexadecimal characters that uniquely identify the unit of work.

member-name
An identifier of one to eight characters that identifies either a member of a partitioned data set (the operating system requires this naming convention) or a member of a data sharing group.

A name for a member of a partitioned data set should not begin with DSN; this can sometimes conflict with DB2-provided member names. If a name beginning with DSN is specified, DB2 issues a warning message.

package-id
An SQL identifier of one to eight letters, digits, or underscores that identifies a package. For packages created under DB2, a package ID is the name of the program whose precompilation produced the package's DBRM. A package ID must begin with a letter. (the operating system requires this naming convention.)

A package ID should not begin with DSN; this can sometimes conflict with DB2-provided package IDs. If a package ID beginning with DSN is specified, DB2 issues a warning message.

For trigger packages, the package ID is the trigger name, which is an SQL identifier of 1 to 128 letters, digits, or underscores that identifies the trigger package.

package-name
A name given to the object created during the bind process of a single package. A package name consists of a location name, a collection ID, and a package ID separated by periods. An additional attribute, a version ID, allows for multiple versions of a package to have the same name.

plan-name
An SQL identifier of one to eight letters, digits or underscores that identifies an application plan. A plan name must begin with a letter.

A plan name should not begin with DSN; this can sometimes conflict with DB2-provided plan names. If a plan name beginning with DSN is specified, DB2 issues a warning message.

qualifier-name
An SQL identifier of 1 to 128 letters, digits, or the underscore that identifies the implicit qualifier for unqualified table names, views, indexes, and aliases.

string
A sequence of characters that begins and ends with an apostrophe.

subsystem-name
An identifier that specifies the DB2 subsystem as it is known to the operating system.

table-name
A qualified or unqualified name that designates a table. A table name can contain one or two parts, depending on its qualification. The first part is the authorization ID that designates the owner of the table; the second part is an SQL identifier. A period must separate each of the parts.

table-space-name
An identifier that designates a table space of an identified database. If a database is not identified, a table space name specifies a table space of database DSNDB04.

utility-id
An identifier of 1 to 16 characters that uniquely identifies a utility process within DB2. A utility ID must begin with a letter, and the identifier can contain periods.

version-id
An SQL identifier of 1 to 64 lowercase alphabetic letters, uppercase alphabetic letters, digits, underscores, at signs (@), number signs (#), dollar signs ($), dashes, and periods that is assigned to a package when the package is created. The version ID that is assigned is taken from the version ID associated with the program being bound. Version IDs are specified for programs as a parameter of the DB2 precompile.

Terminology and citations

In this information, DB2 Universal Database™ for z/OS® is referred to as "DB2 UDB for z/OS." In cases where the context makes the meaning clear, DB2 UDB for z/OS is referred to as "DB2." When this information refers to titles of books in this library, a short title is used. (For example, "See *DB2 SQL Reference*" is a citation to *IBM® DB2 Universal Database for z/OS SQL Reference*.)

When referring to a DB2 product other than DB2 UDB for z/OS, this information uses the product's full name to avoid ambiguity.

The following terms are used as indicated:

DB2 Represents either the DB2 licensed program or a particular DB2 subsystem.

OMEGAMON
Refers to any of the following products:
- IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS
- IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS
- IBM DB2 Performance Expert for Multiplatforms and Workgroups
- IBM DB2 Buffer Pool Analyzer for z/OS

C, C++, and C language
Represent the C or C++ programming language.

CICS® Represents CICS Transaction Server for z/OS or CICS Transaction Server for OS/390®.

IMS™ Represents the IMS Database Manager or IMS Transaction Manager.

MVS Represents the MVS element of the z/OS operating system, which is equivalent to the Base Control Program (BCP) component of the z/OS operating system.

RACF®
Represents the functions that are provided by the RACF component of the z/OS Security Server.

How to read the syntax diagrams

The following rules apply to the syntax diagrams that are used in this book:

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

  The ►►— symbol indicates the beginning of a statement.

  The —► symbol indicates that the statement syntax is continued on the next line.

  The ►— symbol indicates that a statement is continued from the previous line.

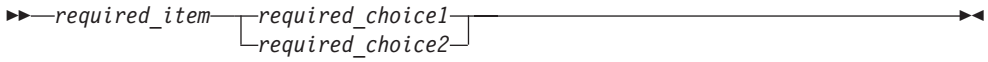
  The —►◄ symbol indicates the end of a statement.
- Required items appear on the horizontal line (the main path).

- Optional items appear below the main path.

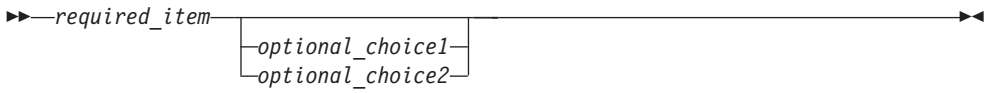
  If an optional item appears above the main path, that item has no effect on the execution of the statement and is used only for readability.

- If you can choose from two or more items, they appear vertically, in a stack.

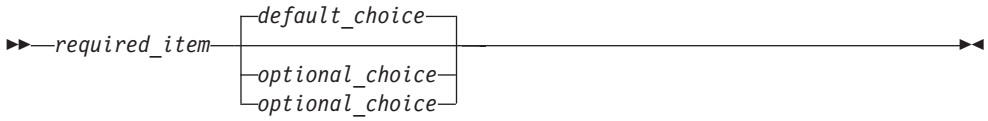
If you *must* choose one of the items, one item of the stack appears on the main path.



If choosing one of the items is optional, the entire stack appears below the main path.

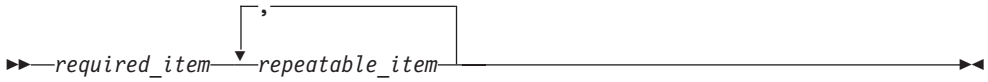


If one of the items is the default, it appears above the main path and the remaining choices are shown below.



- An arrow returning to the left, above the main line, indicates an item that can be repeated.

If the repeat arrow contains a comma, you must separate repeated items with a comma.



A repeat arrow above a stack indicates that you can repeat the items in the stack.

- Keywords appear in uppercase (for example, `FROM`). They must be spelled exactly as shown. Variables appear in all lowercase letters (for example, *column-name*). They represent user-supplied names or values.
- If punctuation marks, parentheses, arithmetic operators, or other such symbols are shown, you must enter them as part of the syntax.

Prerequisite and related information

This book is intended to serve as a reference for people who understand system administration, database administration, or application programming in the DB2 environment. You should have some knowledge of:

- CICS, IMS, or TSO
- z/OS Job Control Language (JCL)
- Structured Query Language (SQL)

Refer to "How to use the DB2 library" on page 445 for information about how to use the DB2 library.

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products. The major accessibility features in z/OS products, including DB2 UDB for z/OS, enable users to:

- Use assistive technologies such as screen reader and screen magnifier software
- Operate specific or equivalent features by using only a keyboard
- Customize display attributes such as color, contrast, and font size

Assistive technology products, such as screen readers, function with the DB2 UDB for z/OS user interfaces. Consult the documentation for the assistive technology products for specific information when you use assistive technology to access these interfaces.

Online documentation for Version 8 of DB2 UDB for z/OS is available in the Information management software for z/OS solutions information center, which is an accessible format when used with assistive technologies such as screen reader or screen magnifier software. The Information management software for z/OS solutions information center is available at the following Web site: http://publib.boulder.ibm.com/infocenter/dzichelp

How to send your comments

Your feedback helps IBM to provide quality information. Please send any comments that you have about this book or other DB2 UDB for z/OS documentation. You can use the following methods to provide comments:

- Send your comments by e-mail to db2zinfo@us.ibm.com and include the name of the product, the version number of the product, and the number of the book. If you are commenting on specific text, please list the location of the text (for example, a chapter and section title or a help topic title).
- You can send comments from the Web. Visit the library Web site at:

  www.ibm.com/software/db2zos/library.html

  This Web site has a an online reader comment form that you can use to send comments.
- You can also send comments by using the feedback link at the footer of each page in the Information Management Software for z/OS Solutions Information Center at http://publib.boulder.ibm.com/infocenter/db2zhelp.

Summary of changes to this book

The principal changes to this book are as follows:

- New options are available for the following commands:
  - -DISPLAY DATABASE (DB2)
  - DSNH (TSO CLIST)
  - MODIFY irlmproc,SET (z/OS IRLM)
  - START irlmproc (z/OS IRLM)
- The REOPT(VARS) and NOREOPT(VARS) options for BIND and REBIND PLAN, PACKAGE, and TRIGGER PACKAGE is changed to REOPT(ALWAYS|NONE) and a third option is added: REOPT(ONCE)
- The following commands have increased functionality:
  - -DISPLAY FUNCTION SPECIFIC (DB2)
  - -DISPLAY DATABASE (DB2)
  - -DISPLAY GROUP (DB2)
  - -DISPLAY PROCEDURE (DB2)
  - DSNH (TSO CLIST)
  - MODIFY irlmproc,SET (z/OS IRLM)
  - MODIFY irlmproc,STATUS (z/OS IRLM)
  - -SET LOG (DB2)
  - -START DATABASE (DB2)
  - -STOP DATABASE (DB2)
- # The following commands are new:
  - # – MODIFY admtproc,APPL=SHUTDOWN (z/OS)
  - # – MODIFY admtproc,APPL=TRACE (z/OS)
  - # – START admtproc (z/OS)
  - # – STOP admtproc (z/OS)

© Copyright IBM Corp. 1983, 2008

Part 1. Privileges, authorization IDs, and the bind process

This part contains information about the privileges and authorization IDs that are required to issue the various commands used for DB2 and a summary of the bind process.

© Copyright IBM Corp. 1983, 2008

Chapter 1. Privileges and authorization IDs

The issuer of a command can be an individual user. It can also be a program running in batch mode or an IMS or CICS transaction. The term *process* is used to represent any or all of those.

A process is represented to DB2 by a set of identifiers (IDs). What the process can do with DB2 is determined by *privileges* and *authorities* that can be held by its identifiers. *"The privilege set of a process"* means the entire set of privileges and authorities that can be used by the process in a specific situation.

There are three types of identifiers: primary authorization IDs, secondary authorization IDs, and SQL IDs.

- Generally the primary authorization ID identifies a specific process. For example, in the process initiated through the TSO attachment facility, the primary authorization ID is identical to the TSO logon ID. A trace record identifies the process by the primary authorization ID.

  If RACF is active, IDs that issue commands from logged-on MVS consoles or from TSO SDSF must have appropriate RACF authorization for DB2 commands, or the primary authorization IDs must have DB2 authorization to issue commands.
- Secondary authorization IDs, which are optional, can hold additional privileges that are available to the process. A secondary authorization ID is often a Resource Access Control Facility (RACF®) group ID. For example, a process can belong to a RACF group that holds the LOAD privilege on a particular database. Any member of the group can run the LOAD utility to load table spaces into the database.

  DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.
- An SQL authorization ID (SQL ID) holds the privileges that are exercised when a process issues certain dynamic SQL statements. This ID does not effect most of the commands that are described in this book.

Within DB2, a process can be represented by a primary authorization ID and possibly by one or more secondary IDs. For detailed instructions on how to associate a process with one or more IDs, and for instructions on how to grant privileges to those IDs, see Part 3 (Volume 1) of *DB2 Administration Guide*.

A privilege or authority is granted to, or revoked from, an identifier by executing an SQL GRANT or REVOKE statement. For the complete syntax of those statements, see Chapter 5 of *DB2 SQL Reference*.

© Copyright IBM Corp. 1983, 2008

Chapter 2. The bind process

The bind process establishes a relationship between an application program and its relational data. This process is necessary before you can execute your program. DB2 allows you two basic ways to bind a program: to a package, or directly to an application plan. If your program uses DRDA access to distribute data, you must use packages.

During the precompilation process, the DB2 precompiler produces both modified source code and a database request module (DBRM) for each application program. The modified source code must be compiled and link-edited before the program can be run. DBRMs must be bound to a plan or package.

When determining the maximum size of a plan, you must consider several physical limitations, including the time required to bind the plan, the size of the EDM pool, and fragmentation. There are no restrictions to the number of DBRMs that can be included in a plan. However, packages provide a more flexible method for handling a large number of DBRMs within a plan. As a general rule, the EDM pool should be at least 10 times the size of the largest DBD or plan, whichever is greater. For further information, see Part 2 of *DB2 Installation Guide*.

The BIND PACKAGE subcommand allows you to bind DBRMs individually. It gives you the ability to test different versions of a program without extensive rebinding. Package binding is also the only method for binding programs at remote sites.

Even when they are bound into packages, all application programs must be designated in an application plan. BIND PLAN establishes the relationship between the DB2 system and all DBRMs or packages in that plan. Plans can specify explicitly named DBRMs, packages, collections of packages, or a combination of these elements. The plan contains information about the designated DBRMs or packages and about the data that the application program intends to use. The plan is stored in the DB2 catalog.

In addition to building packages and plans, the bind process does the following tasks:

- **Validates the SQL statements using the DB2 catalog.** During the bind process, DB2 checks your SQL statements for valid table, view, and column names. Because the bind process occurs as a separate step before program execution, errors are detected and can be corrected before the program is executed.
- **Verifies that the process binding the program is authorized to perform the data accessing operations requested by your program's SQL statements.** When you issue BIND, you can specify an authorization ID as the owner of the plan or package. The owner can be any one of the authorization IDs of the process that is performing the bind. The bind process determines whether the owner of the plan or package is authorized to access the data the program requests.
- **Selects the access paths that are needed to access the DB2 data your program needs to process.** In selecting an access path, DB2 considers indexes, table sizes, and other factors. DB2 considers all indexes that are available to access the data and decides which ones (if any) to use when selecting a path to the data.

© Copyright IBM Corp. 1983, 2008

BIND PLAN and BIND PACKAGE can be accomplished using DB2I panels, the DSNH CLIST, or the DSN subcommands BIND PLAN and BIND PACKAGE. For a detailed explanation of binding with DSNH CLIST, see Part 3, "Commands." A description of the bind process can be found in Part 5 of *DB2 Application Programming and SQL Guide*. Further information on BIND can be found in Chapter 13, "BIND PACKAGE (DSN)," on page 49 and in Chapter 14, "BIND PLAN (DSN)," on page 57. Information about specific options for BIND PLAN and BIND PACKAGE can be found in Chapter 15, "BIND and REBIND options," on page 63.

Part 2. Working with commands

This part contains information about working with commands.

The tables in Chapter 8, “Description of commands,” on page 21 summarize the commands that follow in Part 3. Each table lists commands of one type, describes their functions, and refers to the page on which a complete description begins.

© Copyright IBM Corp. 1983, 2008

Chapter 3. DB2 command parsing

DB2 commands follow a pattern similar to the pattern illustrated in Figure 1:

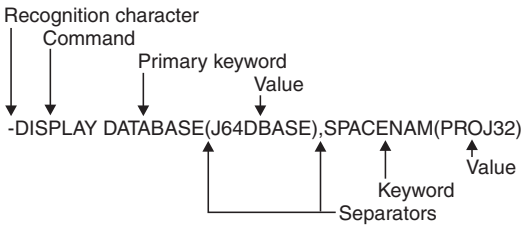


Figure 1. The format and parts of a DB2 command

The following topics provide additional information:

- "Parts of a DB2 command"
- "Characters with special meanings" on page 10
- "Examples of Keyword Entry" on page 11

Parts of a DB2 command

The parts of a command are:

Recognition character

Shown as a hyphen throughout this book, with the following exceptions:

- If the command is issued from a z/OS console, the recognition character must be the *command prefix*.

  In DB2 UDB for z/OS Version 8, the command prefix can be up to eight characters. The default is '-DSN1'. However, the majority of examples in this book assume that the command prefix has been defined as a hyphen (-). Examples involving members of a data sharing group demonstrate the use of multi-character command prefixes, such as -DB1G.

  Inserting a space between the command prefix and the command is optional. For example, you can use either one of the following formats:

  ```
  -DB1GDIS THREAD(*)
  -DB1G DIS THREAD(*)
  ```

  Using a space makes it easier for users to identify the command, especially when the command prefix has multiple characters.

  The command prefix can be defined at installation time. For more information, see Part 2 of *DB2 Installation Guide*.
- If the command is issued from an IMS terminal, the recognition character must be the command recognition character (CRC). The command recognition character is defined in the IMS SSM PROCLIB member. For more information, see *IMS Customization Guide.*
- If the command is issued from a CICS terminal or under the DSN command processor, the recognition character must be a hyphen.

Command name

The name of the command. Command names have abbreviations, which are provided in the each command's description.

© Copyright IBM Corp. 1983, 2008

Operands

Combinations of keywords and parameters that can be specified for the command.

Keywords

Sometimes called command options. Keywords can be required or optional. They must be entered exactly as shown in the descriptions of the commands.

Parameters

A keyword can have zero or more parameters. A parameter list, if present, must be enclosed in parentheses.

Separators

These can be one or more blanks or commas. An open parenthesis marks the beginning of a parameter list; no separator is needed. Optionally, an equal sign can be used to separate a single parameter from its keyword without using parentheses.

Characters with special meanings

The following characters have special meaning for the syntax of DB2 commands:

A blank is a separator.

Multiple blanks are equivalent to a single blank, except in strings enclosed between apostrophes.

, A comma is a separator.

' An apostrophe is the usual SQL string constant delimiter, and marks the beginning or end of a string constant in SQL. (In COBOL programs only, the QUOTESQL precompiler option allows you to choose the quotation mark as the SQL string delimiter; the apostrophe is then the SQL escape character.)

Letters not in string constants are changed to uppercase. Two successive apostrophes in a string constant are changed to one apostrophe. Blanks, commas, equal signs, and parentheses in string constants are treated as literal characters, and are not recognized as separators.

There is an exception to the rule about changing letters to uppercase. If the CODED CHARACTER SET option is set to 930 or 5026 during installation, the letters are not folded to uppercase, whether in an SQL string constant or not.

" A quotation mark is the SQL escape character, and marks the beginning or end of an SQL delimited identifier. (In COBOL programs only, the QUOTESQL precompiler option allows you to choose the apostrophe as the SQL escape character; the double quotation mark is then the SQL string delimiter.)

Within a string delimited by quotation marks, two successive quotation marks are changed to one. Other rules are the same as for SQL string constants.

= An equal sign separates a single parameter from a keyword. Thus, an equal sign is used as a separator for keywords that have only one parameter. An equal sign can be used for keywords with multiple parameters when only one member of the parameter list is specified.

(An open parenthesis marks the beginning of a parameter list.

) A close parenthesis marks the end of a parameter list.

: A colon means an inclusive range. For example, (A:D) means the same as (A,B,C,D); (1:5) means (1,2,3,4,5). The colon can be used this way only in commands where this operation is specifically permitted.

* An asterisk means any of the following usages:

  * A single asterisk as a keyword_value indicates all. For example:

    ```
    -DISPLAY UTILITY (*)
    ```

  ***keyword_value**
  An asterisk as the first character of a keyword_value indicates that a match for the value will be satisfied when all characters following the * are the same. For example: (*BCD)

  keyword*value
  An intermediate asterisk indicates that a match for the value will be satisfied when all characters preceding and all characters following the asterisk are the same. For example: (ABC*EFG)

  keyword_value*
  An asterisk as the final character of a keyword_value indicates that a match will for the value will be satisfied when all characters preceding the asterisk are the same. For example: (ABC*)

  keyword*_value
  Asterisks used as the first, intermediate and final characters in a string are also valid. For example: (*BCD*FGH*)

  For example, DISPLAY UTILITY (*) displays the status of all utilities; DISPLAY UTILITY (R2*) displays the status of all utilities whose identifiers begin with R2.

  The asterisk pattern-matching character is available to all DB2 commands, but not all DB2 commands support an asterisk. The asterisk can be used this way only in commands in which the pattern-matching operation is specifically permitted.

NO (two-character string) negates the keyword that follows.

A negated keyword means the opposite of the keyword itself, and is often used to override a keyword default. In keywords that have no opposite meaning, the initial characters “NO” can be merely part of the keyword itself; for example, in NODE.

Examples of Keyword Entry

General following examples illustrate valid keywords and parameters:
- MODE (FORCE)
- MODE=FORCE
- MODE (NOFORCE) (keyword negation)
- MODE=NOFORCE (keyword negation)
- DATABASE(name1 name2 . . . name*n*) ACCESS(RO)
- SPACENAM (name1,name2) ACCESS(RO)
- ACCESS (RO),SPACENAM=name
- Combinations of the preceding

Do not use more than one parameter after an equal sign, or an error condition occurs.

Chapter 4. Scope of commands

In a data sharing environment, the *scope* of a command is the breadth of its impact. All commands have one of the following scopes:

Member or Local
Many commands used in a data sharing environment have *member* (or *local*) scope because they affect only the DB2 subsystem for which they are issued. For example, a CANCEL THREAD command cancels the specified threads for the member that is identified by the command prefix.

Group Other commands have *group* scope because they affect an object in such a way that affects all members of the group. For example, a STOP DATABASE command issued from any member of the group stops that database for all members of the group.

The following commands have group scope:

ALTER GROUPBUFFERPOOL (DB2)	FREE PLAN (DSN)
BIND PACKAGE (DSN)	MODIFY irlmproc,DIAG (z/OS IRLM)
BIND PLAN (DSN)	REBIND PACKAGE (DSN)
DCLGEN (DSN)	REBIND PLAN (DSN)
DISPLAY DATABASE (DB2)	REBIND TRIGGER PACKAGE (DSN)
DISPLAY GROUP (DB2)	START DATABASE (DB2)
DISPLAY GROUPBUFFERPOOL (DB2)	STOP DATABASE (DB2)
FREE PACKAGE (DSN)	

The following commands have either group or member scope, depending on which options you specify with them:

ARCHIVE LOG (DB2)	START FUNCTION SPECIFIC (DB2)
DISPLAY FUNCTION SPECIFIC (DB2)	START PROCEDURE (DB2)
DISPLAY PROCEDURE (DB2)	START TRACE (DB2)
DISPLAY THREAD (DB2)	STOP FUNCTION SPECIFIC (DB2)
DISPLAY TRACE (DB2)	STOP PROCEDURE (DB2)
DISPLAY UTILITY (DB2)	STOP TRACE (DB2)
MODIFY irlmproc,SET (z/OS IRLM)	TERM UTILITY (DB2)
MODIFY irlmproc,STATUS (z/OS IRLM)	

All other commands have member scope. The description of each command includes its scope. For more details on data sharing, see *DB2 Data Sharing: Planning and Administration*.

© Copyright IBM Corp. 1983, 2008

Chapter 5. Output from DB2 commands

The amount of output that you receive from a DB2 command is always less than 256 KB. The following factors determine the maximum amount of output you can receive:

- The amount of storage available to your DB2 subsystem or to an individual command.
- The environment from which you issue the DB2 command.

  For example, if you issue a DB2 command from an IMS console, you can receive no more than 32 KB of output.
- For DISPLAY DATABASE, the value of the LIMIT parameter.
- For DISPLAY THREAD, the number of lines of output.

  DISPLAY THREAD does not display more than 254 lines of output.

© Copyright IBM Corp. 1983, 2008

Chapter 6. Issuing commands to DB2 from IFI

Consider using IFI to let your programs issue commands to DB2. This method returns information about the success or failure of the command to your program. If the command issues a non-zero return code, the information returned to your program includes diagnostic information about the command processed.

For more information about submitting DB2 commands through IFI, see Appendix E (Volume 2) of *DB2 Administration Guide*.

© Copyright IBM Corp. 1983, 2008

Chapter 7. DSN subcommand parsing

The parsing of DSN subcommands conforms to standard TSO command parsing conventions. For information about TSO command parsing, see *z/OS TSO/E Programming Services.*

To continue a subcommand on the next line while using the DSN processor, type either a hyphen (-) or a plus sign (+) at the end of the current line. If you use a plus sign, precede it by at least one blank character to prevent the concatenation of character strings from line to line. Using a plus sign causes TSO/E to delete leading delimiters (blanks, commas, tabs, and comments) on the continuation line, and will reduce the overall size of the command.

Abbreviations: The names of the DSN command and its subcommands cannot be abbreviated. For compatibility with prior releases of DB2, abbreviations for some keywords are allowed.

Recommendation: To avoid potential problems, avoid abbreviating keywords.

© Copyright IBM Corp. 1983, 2008

Chapter 8. Description of commands

The commands are divided into the following categories:
- "The DSN command and its subcommands"
- "DB2 commands"
- "IMS commands" on page 23
- "CICS attachment facility commands" on page 24
- # "Administrative scheduler commands" on page 24
- "z/OS IRLM commands" on page 25
- "TSO CLISTs" on page 25

The DSN command and its subcommands

Environment: DSN is the DB2 command processor and executes as a TSO command processor. All of its subcommands, except SPUFI, run under DSN in either the foreground or background, and all, except END, also run under DB2 Interactive (DB2I). SPUFI runs only in the foreground under ISPF.

Table 1 lists each DSN command or subcommand described in this book, its function, and the page number where you can find more information about the command.

Table 1. DSN command and subcommands

DSN command or subcommand	Function	Refer to page
BIND	Builds an application package or plan	49, 57
DB2 commands	Execute a DB2 command	Table 2 on page 22
DCLGEN	(DECLARATIONS GENERATOR) Produces declarations for tables or views	105
DSN	Starts a DSN session	217
END	Ends the DSN session	265
FREE	Deletes an application package or plan	267, 271
REBIND	Updates an application package or plan	299, 303
REBIND TRIGGER PACKAGE	Updates an application trigger package	307
RUN	Executes an application program	327
SPUFI	Executes the SQL Processor Using File Input	341

DB2 commands

Environment: The command START DB2 can be issued only from a z/OS console. All other DB2 commands can be issued from the following environments:
- A z/OS console or application program
- A DSN session
- A DB2I panel
- An IMS terminal
- A CICS terminal

© Copyright IBM Corp. 1983, 2008

- An application program, using the DB2 instrumentation facility interface (IFI)

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Extended MCS Consoles: The extended MCS console feature of z/OS lets a z/OS system have more than 99 consoles. Because DB2 supports extended MCS consoles, messages returned from a DB2 command are routed to the extended MCS console that issued the command. For more information on extended MCS consoles, see Part 4 (Volume 1) of *DB2 Administration Guide*, and *z/OS MVS Planning: Operations*.

Table 2 lists each DB2 command, its function, and the page number where you can find more information about the command.

Table 2. DB2 commands

DB2 command	Function	Refer to page
-ALTER BUFFERPOOL	Alters attributes for the buffer pools	29
-ALTER GROUPBUFFERPOOL	Alters attributes for the group buffer pools	35
-ALTER UTILITY	Alters parameter values of the REORG utility	39
-ARCHIVE LOG	Enables a site to close a current active log and open the next available log data set	43
-CANCEL THREAD	Cancels processing for specific local or distributed threads	97
-DISPLAY ARCHIVE	Displays information about archive log processing	119
-DISPLAY BUFFERPOOL	Displays information about the buffer pools	121
-DISPLAY DATABASE	Displays status information about DB2 databases	131
-DISPLAY FUNCTION SPECIFIC	Displays statistics about external user-defined functions	153
-DISPLAY GROUP	Displays information about the data sharing group to which a DB2 subsystem belongs	157
-DISPLAY GROUPBUFFERPOOL	Displays status information about DB2 group buffer pools	163
-DISPLAY LOCATION	Displays status information about distributed threads	177
-DISPLAY LOG	Displays log information and status of the offload task	181
-DISPLAY PROCEDURE	Displays status information about stored procedures	183
-DISPLAY RLIMIT	Displays status information about the resource limit facility (governor)	189
-DISPLAY THREAD	Displays information about DB2 threads	191
-DISPLAY TRACE	Displays information about DB2 traces	205
-DISPLAY UTILITY	Displays status information about a DB2 utility	211
-MODIFY TRACE	Changes the IFCIDs (trace events) associated with a particular active trace	295
-RECOVER BSDS	Reestablishes dual bootstrap data sets	311
-RECOVER INDOUBT	Recovers threads left indoubt	313

Table 2. DB2 commands (continued)

DB2 command	Function	Refer to page
-RECOVER POSTPONED	Completes back-out processing for units of recovery left incomplete during an earlier restart	317
-RESET GENERICLU	Purges information stored by VTAM® in the coupling facility	321
-RESET INDOUBT	Purges information displayed in the indoubt thread report generated by the -DISPLAY THREAD command	323
-SET ARCHIVE	Controls the allocation of tape units and the deallocation time of the tape units for archive log processing	331
-SET LOG	Modifies the checkpoint frequency	335
-SET SYSPARM	Changes subsystem parameters online	339
-START DATABASE	Makes the specified database available for use	351
-START DB2	Initializes the DB2 subsystem (can be issued only from a z/OS console)	361
-START DDF	Starts the distributed data facility	365
-START FUNCTION SPECIFIC	Activates an external function that is stopped	367
-START PROCEDURE	Activates the definition of stopped or cached stored procedures	377
-START RLIMIT	Starts the resource limit facility (governor)	381
-START TRACE	Initiates DB2 trace activity	383
-STOP DATABASE	Makes specified databases unavailable for applications	397
-STOP DB2	Stops the DB2 subsystem	403
-STOP DDF	Stops the distributed data facility	405
-STOP FUNCTION SPECIFIC	Stops the acceptance of SQL statements for specified functions	409
-STOP PROCEDURE	Stops the acceptance of SQL CALL statements for stored procedures	415
-STOP RLIMIT	Stops the resource limit facility (governor)	419
-STOP TRACE	Stops trace activity	421
-TERM UTILITY	Terminates execution of a utility	427

Completion Messages: Message DSN9022I indicates the normal end of DB2 command processing; DSN9023I indicates the abnormal end of DB2 command processing.

IMS commands

Environment:Each IMS command can be issued from an IMS terminal or you can invoke IMS transaction and commands by using the DB2-supplied stored procedure DSNAIMS. For more information about DSNAIMS, see the*DB2 Administration Guide*.

Table 3 on page 24 lists the IMS commands that are described in this book, each command's function, and the page number where you can find more information

about the command

Table 3. IMS commands

IMS command	Function	Refer to page
/CHANGE	Resets an indoubt recovery unit	103
/DISPLAY	Displays the status of the connection between IMS and the specified subsystem (DB2), or displays the outstanding recovery units associated with the specified subsystem (DB2)	115
/SSR	Allows the IMS operator to enter an external subsystem (DB2) command	343
/START	Makes available the connection between IMS and the specified external subsystem (DB2)	349
/STOP	Prevents application programs from accessing the external subsystem's (DB2's) resources	395
/TRACE	Allows users to direct and control IMS tracing activities	431

CICS attachment facility commands

Environment: Each CICS attachment facility command can be issued from a CICS terminal.

Table 4 lists the CICS attachment facility commands described in this book, each command's function, and the page number where you can find more information about each command.

Table 4. CICS attachment facility commands

CICS attachment facility commands	Function	Refer to page
DSNC	Allows you to enter DB2 commands from CICS	221
DSNC DISCONNECT	Disconnects threads	223
DSNC DISPLAY	Displays information on CICS transactions	225
DSNC MODIFY	Modifies the message queue destination of the DB2CONN, or modifies the maximum active thread value for the pool, for DSNC commands, or for DB2ENTRY	229
DSNC STOP	Stops the CICS attachment facility	231
DSNC STRT	Starts the CICS attachment facility	233

Administrative scheduler commands

You can use administrative scheduler commands to start, stop, and change the administrative scheduler.

Environment: Each administrative scheduler command can be issued from a z/OS console.

Table 5. Administrative scheduler commands

z/OS command	Function	Refer to page
MODIFY admtproc,APPL=SHUTDOWN	Stops the administrative scheduler from accepting new requests and starting new task executions. It also shuts down the administrative scheduler.	273
MODIFY admtproc,APPL=TRACE	Starts or stops traces in the administrative scheduler.	275
START admtproc	Starts the scheduler that is specified in the *admtproc* parameter.	345
STOP admtproc	Starts the scheduler that is specified in the *admtproc* parameter.	347

z/OS IRLM commands

Environment: Each z/OS IRLM command can be issued from a z/OS console.

Table 6 lists each z/OS IRLM command described in this book, each command's function, and the page number where you can find more information about each command.

Table 6. z/OS commands affecting the IRLM

z/OS command	Function	Refer to page
MODIFY irlmproc,ABEND	Abends IRLM	277
MODIFY irlmproc,DIAG	Initiates diagnostic dumps for IRLM subsystems	279
MODIFY irlmproc,PURGE	Releases IRLM retained locks	281
MODIFY irlmproc,SET	Dynamically sets various IRLM operational parameters	283
MODIFY irlmproc,STATUS	Displays IRLM status	287
START irlmproc	Starts an IRLM component with an installation-supplied procedure	371
STOP irlmproc	Shuts down IRLM normally	413
TRACE CT	Starts, stops, or modifies IRLM tracing	433

TSO CLISTs

Table 7 lists the TSO CLIST command, each command's function, and, in the case of DSNH, the page number where you can find more information about this command.

Table 7. TSO CLISTs

CLIST	Function	Refer to page
DSNH	Prepares a program for execution, and executes it if it runs under TSO. Runs under TSO in foreground or background.	235

Table 7. TSO CLISTs (continued)

CLIST	Function	Refer to page
DSNU	Generates JCL to execute DB2 utility jobs. Can be executed directly or by using DB2I. For details about this command procedure, see *DB2 Utility Guide and Reference*.	

Part 3. Commands

This part contains syntax diagrams, semantic descriptions, rules, and usage examples of commands, organized alphabetically by command name.

© Copyright IBM Corp. 1983, 2008

Chapter 9. -ALTER BUFFERPOOL (DB2)

The DB2 command ALTER BUFFERPOOL alters attributes for active or inactive buffer pools. Altered values are used until altered again.

Abbreviation: -ALT BPOOL

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 30
- "Option descriptions" on page 30
- "Usage notes" on page 32
- "Examples" on page 34

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS® terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

Authorization

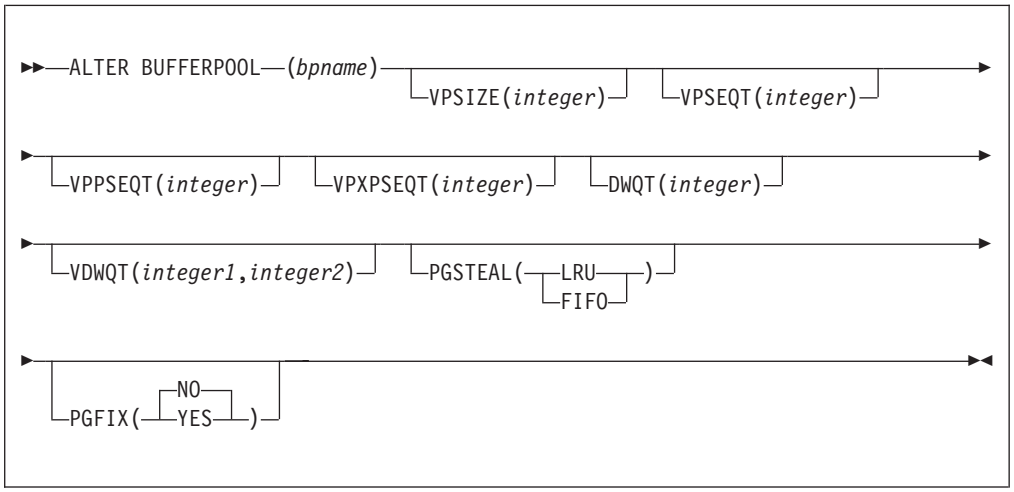
To execute this command, you must use a privilege set of the process that includes one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

(*bpname***)**
Specifies the buffer pool to alter.
- 4-KB page buffer pools are named BP0, BP1, ..., BP49.
- 8-KB page buffer pools are named BP8K0, BP8K1, ..., BP8K9.
- 16-KB page buffer pools are named BP16K0, BP16K1, ..., BP16K9.
- 32-KB page buffer pools are named BP32K, BP32K1, ..., BP32K9.

VPSIZE (*integer***)**
Changes the buffer pool size.

integer specifies the number of buffers to allocate to the active buffer pool.

integer can range from 0 to 250000000 for 4-KB page buffer pools other than BP0. For BP0, the minimum value is 2000. For 8-KB page buffer pools, the range is from 1000 to 125000000. For 16-KB page buffer pools, the range is from 0 to 62500000. For 32-KB page buffer pools, the range is from 0 to 31250000.

DB2 limits the total VPSIZE for all buffer pools to 1 TB. In addition, DB2 limits the amount of buffer pool storage to approximately twice the available real storage for the z/OS image.

If you specify VPSIZE as 0 for an active buffer pool (other than BP0), DB2 quiesces all current database access and update activities for that buffer pool and then deletes the buffer pool. Subsequent attempts to use table spaces or indexes that are assigned to that buffer pool fail.

VPSEQT (*integer***)**
Changes the sequential steal threshold for the buffer pool.

integer specifies the sequential steal threshold for the buffer pool. This value is expressed as a percentage of the total buffer pool size, and valid values range from 0 to 100. This threshold affects the allocation of buffers in the buffer pool to page read requests that are part of a sequential access pattern. This includes pages being prefetched. If the number of buffers that contain sequentially accessed pages exceeds the threshold, a sequential request attempts to reuse

one of those buffers rather than a buffer that contains a non-sequentially accessed page. The initial default value is 80.

When VPSEQT=0, sequentially accessed pages are not kept in the buffer pool after being released by the accessing agent. Also, prefetch is disabled.

When VPSEQT=100, DB2 does not prefer reusing sequential buffers over using non-sequential buffers.

VPPSEQT (*integer***)**
Changes the parallel sequential threshold for the buffer pool. This threshold determines how much of the buffer pool is used for parallel processing operations.

integer specifies the parallel sequential threshold for the buffer pool. This value is expressed as a percentage of the sequential steal threshold, and valid values range from 0 to 100. The initial default value is 50.

When VPPSEQT=0, parallel processing operations are disabled.

VPXPSEQT (*integer***)**
Changes the assisting parallel sequential threshold for the buffer pool. This threshold determines the portion of the buffer pool that is used for processing queries that originate on other members of the data sharing group. This option is valid and effective only when DB2 is in data sharing mode; it is ignored when DB2 is not in data sharing mode.

integer specifies the assisting parallel sequential threshold for the buffer pool. *integer* is expressed as a percentage of the parallel sequential threshold (VPPSEQT). Whenever the sequential steal threshold or the parallel sequential threshold is altered, it directly affects the portion of buffer resources dedicated to assistant parallel operations. The valid values range from 0 to 100. The initial default value is 0.

When VPXPSEQT=0, this buffer pool cannot be used to assist another DB2 with parallel processing.

DWQT (*integer***)**
Changes the buffer pool's deferred write threshold.

integer specifies the deferred write threshold for the buffer pool. This value is expressed as a percentage of the total buffer pool size, and valid values range from 0 to 90. This threshold determines when deferred writes begin, based on the number of unavailable buffers. When the count of unavailable buffers exceeds the threshold, deferred writes begin. The initial default value is 30 percent.

VDWQT (*integer1***,***integer2***)**
Changes the buffer pool's vertical deferred write threshold.

integer1 specifies the vertical deferred write threshold for the buffer pool. *integer1* is expressed as a percentage of the total buffer pool size, and valid values range from 0 to 90.

This threshold determines when deferred writes begin, based on the number of updated pages for a given data set. Deferred writes begin for that data set when the count of updated buffers for a data set exceeds the threshold. This threshold can be overridden for page sets accessed by DB2 utilities and must be less than or equal to the value specified for the DWQT option.

The default value is 5 percent. A value of 0 indicates that the deferred write of 32 pages begins when the updated buffer count for the data set reaches 40.

integer2 specifies the vertical deferred write threshold for the buffer pool. *integer2* is expressed as an absolute number of buffers. You can use *integer2* when you want a relatively low threshold value for a large buffer pool, but *integer1* cannot provide a fine enough granularity between *integer1* values of 0 and 1. *integer2* only applies when *integer1* is 0; DB2 ignores a value specified for *integer2* if the value specified for *integer1* is non-zero. *integer2* can range from 0 to 9999. The default value is 0.

If *integer1* is 0 and *integer2* is a non-zero value, DB2 uses the value specified for *integer2* to determine the threshold. If both values are 0, the *integer1* value of 0 is used as the threshold.

PGSTEAL
Specifies the page-stealing algorithm that DB2 uses for the buffer pool.

The initial default is PGSTEAL (LRU). However, when you issue the ALTER BUFFERPOOL command, you must explicitly specify either LRU or FIFO when the PGSTEAL option is used.

(LRU)
Specifies that the buffer pool buffers should be managed by using the least recently used (LRU) algorithm.

(FIFO)
Specifies that the buffer pool buffers should be managed by using the first-in-first-out (FIFO) algorithm.

PGFIX
Specifies whether the buffer pool should be fixed in real storage when it is used.

(NO)
Specifies that the buffer pool is not fixed in real storage. Page buffers are fixed and unfixed in real storage across each I/O and group buffer pool operation.

This is the default.

(YES)
Specifies that the buffer pool is fixed in real storage. Page buffers are fixed when they are first used after the buffer pool is allocated or expanded.

Usage notes

The following sections contain additional information about how to use the ALTER BUFFERPOOL command.

Changing several buffer pool attributes: A failure in modifying one buffer pool attribute has no effect on other modifications requested in the same command.

Insufficient virtual storage: If insufficient virtual storage is detected while expanding a buffer pool, DB2 issues an error message, and the process terminates, leaving the buffer pool with a smaller size than was requested.

Contracting an active buffer pool: If you use ALTER BUFFERPOOL to contract the size of an active buffer pool, DB2 contracts the pool by marking active buffers as "to be deleted," which means that they are not reusable to satisfy other page requests. However, the virtual storage might not be freed immediately. A system administrator can determine the status of the buffer pool by issuing the DISPLAY BUFFERPOOL command.

Deleting an active buffer pool: If you use ALTER BUFFERPOOL to delete an active buffer pool (by specifying 0 for VPSIZE), DB2 issues a message to indicate that it is ready to explicitly delete this buffer pool. When DB2 accepts the delete buffer pool request, the buffer pool is marked as "delete pending". All current access to the buffer pool is quiesced, later access attempts fail with an error message, and all open page sets that refer to the buffer pool are closed.

Altering attributes stored in the BSDS: The buffer pool attributes that are stored in the BSDS cannot be changed offline.

Setting a buffer pool to be fixed in real storage: If you use the ALTER BUFFERPOOL command with the PGFIX option set to YES to fix a buffer pool in real storage, the change is pending and the buffer pool becomes fixed only at the next allocation.

In order to fix the buffer pool in real storage, issue the command `ALTER BUFFERPOOL(bpname) PGFIX(YES)`. If the buffer pool that you specify for *bpname* is not currently allocated, the buffer pool will become fixed in real storage when it is allocated. If the buffer pool that you specify for *bpname* is currently allocated, do one of the following procedures to fix the buffer pool in real storage:

- If the buffer pool that you specify for *bpname* is **not** one of the buffer pools that is used for the DB2 catalog and directory (BP0, BP8K0, BP16K0, or BP32K):
  1. Issue the ALTER BUFFERPOOL command with the VPSIZE option set to 0 to deallocate the buffer pool:

     ```
     -ALTER BUFFERPOOL(bpname) VPSIZE(0)
     ```

  2. Issue the ALTER BUFFERPOOL command with the VPSIZE and PGFIX options to change the buffer pool size and to use long-term page fixing at the next allocation:

     ```
     -ALTER BUFFERPOOL(bpname) VPSIZE(vpsize) PGFIX(YES)
     ```

- If the buffer pool that you specify for *bpname* is one of the buffer pools that is used for the DB2 catalog and directory (BP0, BP8K0, BP16K0, or BP32K):
  1. Issue the ALTER BUFFERPOOL command with the PGFIX option to change the buffer pool to use long-term page fixing (the change is pending until the next allocation of the buffer pool):

     ```
     -ALTER BUFFERPOOL(bpname) PGFIX(YES)
     ```

  2. Issue the STOP DATABASE command or the STOP DB2 command to deallocate the buffer pool
  3. Issue the START DATABASE command or the START DB2 command to reallocate the buffer pool (depending on which command you used to deallocate the buffer pool)

Relating VPPSEQT and VPXSEQT: Table 8 on page 34 explains how the two parallel sequential thresholds, VPPSEQT for parallel sequential and VPXPSEQT for assisting parallel sequential threshold, are related. VPXPSEQT is a percentage of VPPSEQT, which is itself a portion of VPSEQT. Multiply VPXPSEQT by VPPSEQT to obtain the total amount of the buffer pool that can be used to assist another DB2 subsystem with parallel processing. In addition, VPPSEQT is affected by changing VPSIZE and VPSEQT; therefore, VPXPSEQT is also affected by VPSIZE and VPSEQT. For more information about the relationships of the various thresholds and possible configurations, see Chapter 6 of *DB2 Data Sharing: Planning and Administration*.

Table 8. Relationship between VPPSEQT and VPXPSEQT

If VPPSEQT is set to	and VPXPSEQT is set to	The percentage of the buffer pool available to assist Sysplex query parallelism equals
50	50	25
50	100	50
100	50	50
any value	0	0
0	any value	0

Examples

Example 1: The following command sets the buffer pool for BP0 to 2000.

```
-ALTER BUFFERPOOL(BP0) VPSIZE(2000)
```

Example 2: The following command sets the sequential steal threshold of the buffer pool for BP0 to 75 % of the buffer pool size.

```
-ALTER BUFFERPOOL(BP0) VPSEQT(75)
```

Example 3: The following command deletes BP1. Be very careful when using this option because specifying a 0 size for an active buffer pool causes DB2 to quiesce all current database access. All subsequent requests to open page sets fail.

```
-ALTER BUFFERPOOL(BP1) VPSIZE(0)
```

Chapter 10. -ALTER GROUPBUFFERPOOL (DB2)

The DB2 command ALTER GROUPBUFFERPOOL alters attributes of group buffer pools.

Abbreviation: -ALT GBPOOL

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions" on page 36
- "Usage notes" on page 37
- "Examples" on page 38

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group

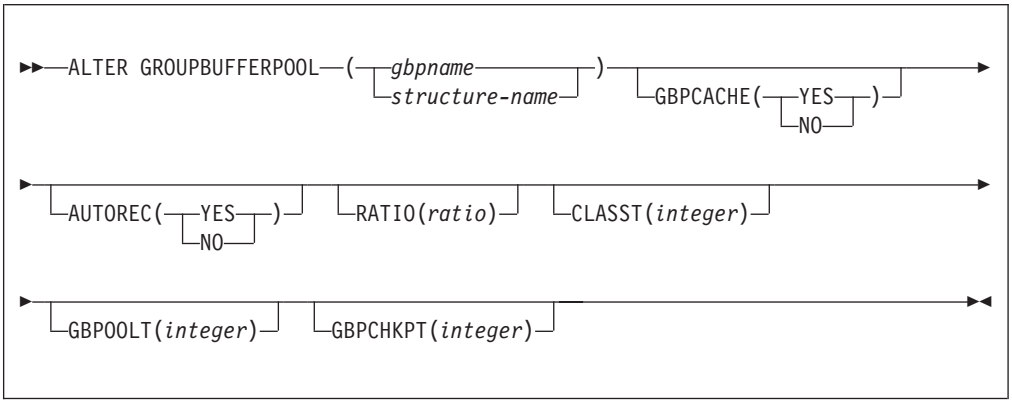
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



© Copyright IBM Corp. 1983, 2008

Option descriptions

(*gbpname***)**
Specifies the DB2 group buffer pool to alter.
- 4-KB group buffer pools are named GBP0, GBP1, ... , GBP49.
- 8-KB group buffer pools are named GBP8K0, GBP8K1, ... , GBP8K9.
- 16-KB group buffer pools are named GBP16K0, GBP16K1, ... , GBP16K9.
- 32-KB group buffer pools are named GBP32K, GBP32K1, ... , GBP32K9.

(*structure-name***)**
Specifies the coupling facility structure for the group buffer pool. The coupling facility structure name has the format, `groupname_gbpname`.

groupname is the DB2 data sharing group name and the underscore (_) separates *groupname* and *gbpname*.

GBPCACHE
Specifies whether *gbpname* is to be used for both caching data and cross-invalidation, or just for cross-invalidation.

(YES)
Indicates that *gbpname* is used for caching data and cross-invalidation.

Any no-data-caching attribute that is specified at either the page set or group buffer pool level takes precedence over a caching specification. For more information, refer to Table 9.

Table 9. Precedence of a no-data-caching specification

Group buffer pool specification	Page set specification	Attribute that takes precedence
GBPCACHE(NO)	GBPCACHE CHANGED GBPCACHE ALL	GBPCACHE(NO)
GBPCACHE(YES)	GBPCACHE NONE	GBPCACHE NONE

(NO)
Indicates that *gbpname* is used only for cross-invalidation. This group buffer pool contains no data entries. The GBPCACHE option of table spaces or index spaces that use this group buffer pool is ignored.

AUTOREC
Specifies whether automatic recovery by DB2 takes place when a structure failure occurs or when the connectivity of all members of the group to the group buffer pool is lost.

(YES)
Enables DB2 to automatically recover page sets and partitions that have a status of group buffer pool RECOVER pending (GRECP) and that have pages on the logical page list.

(NO)
Disables automatic recovery. Issue a START DATABASE command to recover page sets and partitions that have a status of GRECP or that have pages on the logical page list.

RATIO (*ratio***)**
Changes the desired ratio of the number of directory entries to the number of data pages in the group buffer pool; that is, how many directory entries exist for each data page.

ratio can be a decimal number from 1.0 to 255, inclusive. Any digits after the first decimal place are ignored; for example, 5.67 is treated as 5.6. If *ratio* is greater than 25, any digits after the decimal point are ignored; for example, 25.98 is treated as 25. The default ratio is 5.

The actual number of directory entries and data pages that are allocated depends on the size of the coupling facility structure, which is specified in the coupling facility policy definitions (CFRM policy).

CLASST (*integer***)**
Changes the threshold at which class castout is started. *integer* is expressed as a percentage of the number of data entries; it can range from 0 to 90. The default is 5 percent.

For example, CLASST(10) starts class castout when the number of pages in that class equals 10 % of the group buffer pool page capacity.

GBPOOLT (*integer***)**
Changes the threshold at which data in the group buffer pool is cast out to disk. *integer* is expressed as a percentage of the number of data entries and can range from 0 to 90. The default is 30 percent.

For example, GBPOOLT(55) casts out data if the number of pages in the group buffer pool equals 55 % of the group buffer pool page capacity.

GBPCHKPT (*integer***)**
Changes the time interval, in minutes, between successive checkpoints of the group buffer pool. *integer* can range from 1 to 999999. Unless a value is explicitly specified for the GBPCHKPT option, the default value is 4 minutes.

The more frequently checkpoints are taken, the less time it takes to recover the group buffer pool if the coupling facility fails.

Usage notes

Defaults: Issuing the ALTER GROUPBUFFERPOOL command does not change any option that is not explicitly specified; the default is to leave the value unchanged. Table 10 lists the default values for the options when the command is first issued for a group buffer pool or a structure.

Table 10. Default option values when ALTER GROUPBUFFERPOOL is first issued

Option	Value
GBPCACHE	YES
RATIO	5
CLASST	5 (%)
GBPOOLT	30 (%)
GBPCHKPT	4 (minutes)

When new values take effect: When you issue the ALTER GROUPBUFFERPOOL command, some option specifications become effective only at the next allocation of the group buffer pool. Table 11 on page 38 lists each option, when the new value takes effect, and if the option is applicable for a group buffer pool that is specified as GBPCACHE(NO).

Table 11. Changing group buffer pool attributes

Keyword	New value takes effect	Applicable if GBPCACHE(NO)?
GBPCACHE	at next allocation	N/A
AUTOREC	immediately	No
RATIO	at next allocation [2]	No [3]
CLASST	immediately	No [3]
GBPOOLT	immediately	No [3]
GBPCHKPT	immediately	No [3]

Notes:

1. You can use the z/OS command SETXCF START,REBUILD to have the change take effect if the group buffer pool is not duplexed. If the group buffer pool is duplexed and you want to change to GBPCACHE(NO), first go back to simplex mode and rebuild. GBPCACHE(NO) is not allowed for duplexed group buffer pools.
2. You can use the z/OS command SETXCF START,REBUILD to have the change take effect if the group buffer pool is not duplexed. If the group buffer pool is duplexed, first go back to simplex mode and rebuild; then optionally go back to duplex mode. If a group buffer pool is duplexed, both instances of that duplexed group buffer pool use the same RATIO value.
3. DB2 issues message DSNB761 when you specify this option for a GBPCACHE(NO) group buffer pool. These settings only take effect after the GBPCACHE attribute has been changed to YES.

Examples

Example 1: For group buffer pool 0, change the ratio of directory entries to data pages to one directory entry for every data page. The RATIO specification becomes effective at the next allocation of the group buffer pool.

```
-DB1G ALTER GROUPBUFFERPOOL (GBP0) RATIO(1)
```

Example 2: For group buffer pool 2, change the class castout threshold to 10 % and the group buffer pool castout threshold to 50 %. The new values take effect immediately.

```
-DB1G ALTER GROUPBUFFERPOOL (GBP2) CLASST(10) GBPOOLT(50)
```

Example 3: Assume that the DB2 group name is DSNCAT. For group buffer pool 3, change the class castout threshold to 10 %. The new value takes effect immediately. Because the group name is DSNCAT, the coupling facility structure name is DSNCAT_GBP3. Also, when a structure fails, the AUTOREC(YES) option enables DB2 to automatically recover the page sets and partitions that are in a GRECP status or that have pages on the logical page list.

```
-DB1G ALTER GROUPBUFFERPOOL (DSNCAT_GBP3) CLASST(10) AUTOREC(YES)
```

Example 4: For group buffer pool 32K, change the GBP checkpoint frequency to five minutes. The new value takes effect immediately. In this example, with AUTOREC(NO) specified, DB2 does not start automatic recovery when a structure fails. You might choose this option if you want to determine what page sets or partitions are in a GRECP status or have pages on the logical page list before you enter the START DATABASE command to recover the data with the options you specify.

```
-DB1G ALTER GROUPBUFFERPOOL (GBP32K) GBPCHKPT(5) AUTOREC(NO)
```

Chapter 11. -ALTER UTILITY (DB2)

The DB2 command ALTER UTILITY changes the values of certain parameters of an execution of the REORG utility that uses SHRLEVEL REFERENCE or CHANGE. Specifically, this command changes the values of DEADLINE, MAXRO, LONGLOG, and DELAY. For more information about those parameters and the REORG utility, see *DB2 Utility Guide and Reference*.

REORG can be altered only from the DB2 on which it is running.

Abbreviation: -ALT UTIL

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 40
- "Option descriptions" on page 40
- "Usage note" on page 41
- "Example" on page 41

Environment

This command can be issued from a z/OS console, a DSN session, a DB2I panel (DB2 COMMANDS), an IMS or a CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

Authorization

To execute this command, you must use the primary or some secondary authorization ID of the process that originally submitted the utility job. Alternatively, you must use a privilege set of the process that includes one of the following authorities:

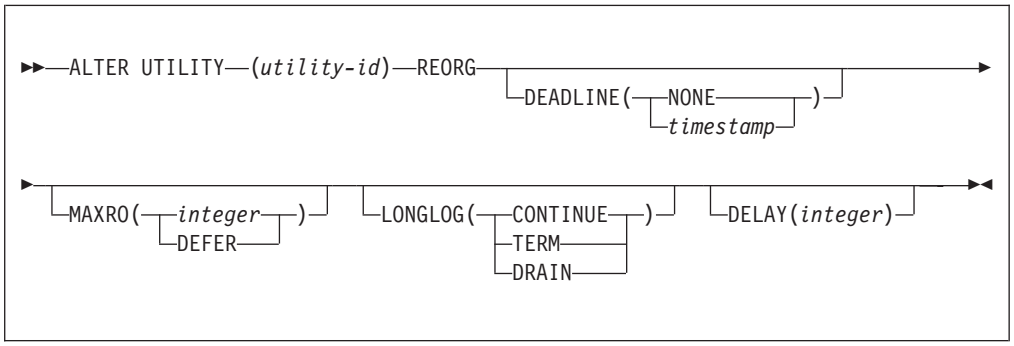
- DBMAINT authority
- DBCTRL authority
- DBADM authority
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

For users with DBMAINT, DBCTRL, or DBADM authority, the command takes effect only when a user has sufficient authority over each object that the utility job accesses.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

(*utility-id***)**
Is the utility identifier, or the UID parameter, used when creating the utility job step.

This job must execute REORG with SHRLEVEL CHANGE or SHRLEVEL REFERENCE.

If *utility-id* was created by the DSNU CLIST by default, it has the form *tso-userid.control-file-name*. For the control file name that is associated with each utility, see *DB2 Utility Guide and Reference*.

If *utility-id* was created by default by the EXEC statement that executed DSNUTLIB, it has the form *userid.jobname*.

DEADLINE
Specifies the deadline by which the user wants the switch phase of reorganization to start. If DB2 estimates that the switch phase will not start by the deadline, DB2 terminates reorganization. The default is the most recently specified value of DEADLINE.

The pre-switch processing might continue until after the deadline.

(NONE)
Specifies that there is no deadline for the read-only iteration of log processing.

(*timestamp***)**
Specifies the deadline by which the user wants the switch phase to start processing. This deadline must not have been reached when ALTER UTILITY executes. For more information on the format for specifying a timestamp, see the discussion of data types in *DB2 SQL Reference*.

MAXRO
Specifies the maximum amount of time that is tolerated for the last iteration of log processing during reorganization. During that iteration, applications have read-only access.

The actual execution time of the last iteration can exceed the value specified for MAXRO.

(*integer***)**
Specifies the number of seconds. The default is the most recently specified value of MAXRO.

(DEFER)
Specifies that the log phase is deferred indefinitely.

LONGLOG
Specifies the action that DB2 performs (after sending the LONGLOG message to the console) if the number of log records that are processed during the next iteration is not sufficiently lower than the number of log records that were processed during the previous iterations. The default is the most recently specified value of LONGLOG.

(CONTINUE)
Specifies that DB2 continues performing reorganization.

(TERM)
Specifies that DB2 terminates reorganization after the delay.

(DRAIN)
Specifies that DB2 drain the write claim class after the delay (if specified). The number of log records, and thus the **estimated** time, for a future iteration of log processing will be 0.

DELAY (*integer***)**
Specifies a lower bound for the interval between the time when REORG sends the LONGLOG message to the console and the time when REORG performs the action specified by the LONGLOG parameter.

integer is the delay in seconds. The value must be nonnegative. The default is the most recently specified value of DELAY.

Usage note

You can alter a REORG job only from the DB2 subsystem on which it is running.

Example

The following example alters the execution of the REORG utility for the utility job step whose utility identifier is REORGEMP:

```
-ALTER UTILITY (REORGEMP) REORG MAXRO(240) LONGLOG(DRAIN)
```

The following list explains what each option does in the preceding example:

- MAXRO(240) changes the maximum tolerable time for the last iteration of log processing to 240 seconds (4 minutes).
- LONGLOG(DRAIN) specifies that DB2 drain the write claim class (if reading of the log during REORG is not catching up to the speed at which the application is writing the log).
- DELAY is not specified so this example does not change the existing delay between sending the LONGLOG message to the console and performing the action specified by LONGLOG.
- DEADLINE is not specified so this example does not change the deadline (if any) that was defined in the last iteration of log processing.

Chapter 12. -ARCHIVE LOG (DB2)

When issued without any options, the DB2 command ARCHIVE LOG performs the following functions:

- Truncates the current active log data sets
- Starts an asynchronous task to offload the data sets
- Archives previous active log data sets not yet archived
- Returns control to the user (immediately)

In a data sharing environment, you can truncate and archive the logs for an individual member or for all members in the group.

When specified with the option MODE(QUIESCE), the ARCHIVE LOG command attempts to quiesce (suspend) all DB2 user update activity on the DB2 active log prior to the offload process. When a system-wide point of consistency is reached (that is, when all currently active update users have reached a commit point), the active log is immediately truncated, and the offload process is initiated. The resulting point of consistency is captured in the current active log before it is off-loaded. In a data sharing environment, you can create a system-wide point of consistency only for the entire group.

For more information regarding the ARCHIVE LOG command, see Part 4 (Volume 1) of *DB2 Administration Guide*.

Abbreviation: -ARC LOG

The following topics provide additional information:

- "Environment"
- "Authorization" on page 44
- "Syntax" on page 44
- "Option descriptions" on page 44
- "Usage notes" on page 46
- "Examples" on page 47

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

The ARCHIVE LOG command can also be issued from the z/OS subsystem interface (SSI) to enable automated scheduling systems and other programs to execute the command via supervisor call instruction (SVC) 34.

Data sharing scope: Group or member, depending on whether you specify MODE(QUIESCE), or on which SCOPE option you choose

© Copyright IBM Corp. 1983, 2008

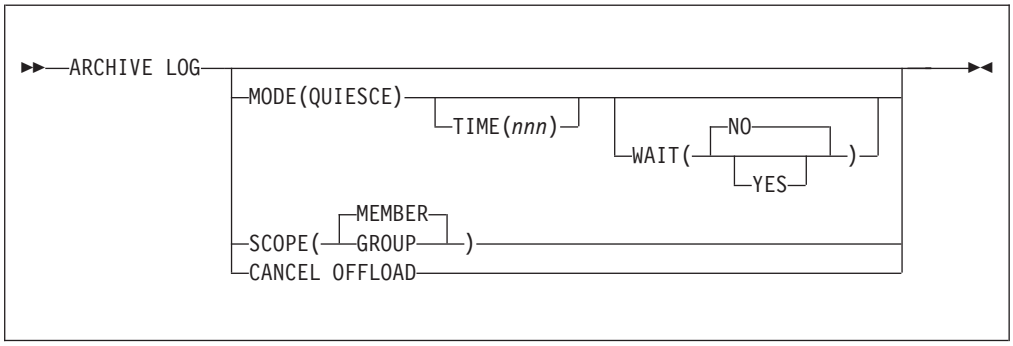
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- ARCHIVE privilege
- Installation SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Option descriptions

MODE(QUIESCE)

Halts all new update activity by the DB2 subsystem for a specified period of time and attempts to bring all existing users to a point of consistency after a commit or rollback. When a point of consistency is reached and captured in the current active log data set, the current active log data set is truncated, and another log data set in the inventory becomes current. Offload processing then begins with the oldest active log data set and ends with the active log data set that was truncated.

In a data sharing environment, before archiving logs of any member, this option quiesces all active members of a data sharing group. MODE(QUIESCE) also ensures that each inactive member had successfully quiesced its update activity and resolved any indoubt units of recovery (URs) before the inactive subsystem completed normal termination. If any DB2 subsystem is in a failed state, fails during quiesce processing, or is stopped with outstanding URs, the ARCHIVE LOG command fails, and the remaining active members allow update activity to proceed.

If no indoubt URs exist on all quiesced members, active or inactive, the archive operation can continue for active members in the group. Thus, you can archive logs of a data sharing group normally without forcing all members to be active. The current logs of inactive members are truncated and off-loaded after they start.

If a system-wide point of consistency cannot be reached during the quiesce period, which is a length of time you can specify, execution of the ARCHIVE LOG command fails and an error message is issued. In a data sharing

environment, the maximum time period applies for the whole group, and if any DB2 subsystem cannot quiesce within the time allowed, the command fails.

If there is no update activity on DB2 data when the command ARCHIVE LOG MODE(QUIESCE) is issued, the active log is truncated and off-loaded immediately.

TIME(*nnn***)**
Specifies the maximum length of time, in seconds, in which the DB2 subsystem is allowed to attempt a full system quiesce.

If you do not specify a time, the **default** is the length of time specified in the field QUIESCE PERIOD of installation panel DSNTIPA. See Part 2 of *DB2 Installation Guide* for more information about this field.

nnn can range from 001 to 999 seconds. You must allocate an appropriate time period for the quiesce processing or the following events can occur:
- The quiesce processing can expire before a full quiesce is accomplished.
- An unnecessary DB2 lock contention can be imposed.
- A time-out can occur.

This option is valid only when used in conjunction with the option MODE(QUIESCE).

WAIT
Specifies whether the DB2 subsystem should wait until the quiesce processing has completed before returning control to the invoking console or program, or should return control when the quiesce processing begins.

This option is valid only when used in conjunction with the option MODE(QUIESCE).

(NO)
Specifies that control must be returned to the invoking program when the quiesce processing begins.

If WAIT(NO) is used, quiesce processing is asynchronous to the user; that is, you can issue additional DB2 commands after the ARCHIVE LOG command returns control to you.

(YES)
Specifies that the quiesce processing must complete before returning control to the invoking console or program.

If WAIT(YES) is used, quiesce processing is synchronous to the user; that is, additional DB2 commands can be issued, but they are not processed by the DB2 command processor until the ARCHIVE LOG command is complete.

SCOPE
Specifies whether the command applies to the entire data sharing group or to a single member only. The SCOPE option is valid only in a data sharing environment; the option is ignored in a non-data-sharing environment. SCOPE cannot be specified if MODE(QUIESCE) is specified; the two keywords are mutually exclusive.

(MEMBER)
Initiates offload processing only for the member from which the command is issued. User update activity is not suspended. If that member, or the entire group, is already archiving, the command fails. This is the default, except when MODE(QUIESCE) is specified.

(GROUP)
Initiates offload processing for every member of the DB2 group. User update activity is not suspended. If any member of the group, or the entire group, is already archiving, the command fails.

CANCEL OFFLOAD
Cancels any off loading currently in progress and restarts the off-load process, beginning with the oldest active log data set that has not been off loaded and proceeding through all active log data sets that need off loading. Any suspended off-load operations are restarted.

Usage notes

Remote site recovery: The ARCHIVE LOG command is very useful when performing a DB2 backup in preparation for a remote site recovery. For example, the command allows the DB2 subsystem to quiesce all users after a commit point, and capture the resulting point of consistency in the current active log *before* the archive is taken. Therefore, when the archive log is used with the most current image copy (during an offsite recovery), the number of data inconsistencies will be minimized. See Part 4 (Volume 1) of *DB2 Administration Guide* for additional information on backup and recovery.

Simultaneous executions: The ARCHIVE LOG command cannot be executed if another ARCHIVE LOG command is in progress. Instead, error message DSNJ318I is issued and the command fails. This is true in both data sharing and non-data-sharing environments. For example, in a data sharing environment, the command fails if the data sharing member, or group to which it belongs, is already archiving.

Available active log space: ARCHIVE LOG cannot be used when the current active log is the last available active log data set because of the following reasons:
- All available active log space would be used.
- The DB2 subsystem would halt processing until an offload is complete.

Executing ARCHIVE LOG while STOP DB2 is in progress: ARCHIVE LOG without the option MODE(QUIESCE) is permitted when STOP DB2 MODE(QUIESCE) is in progress. However, if an attempt is made to execute the ARCHIVE LOG command when a STOP DB2 MODE(FORCE) is in progress, error message DSNJ315I is issued and the ARCHIVE LOG command is not processed.

ARCHIVE LOG with the option MODE(QUIESCE) is not allowed when a STOP DB2 MODE(FORCE) or STOP DB2 MODE(QUIESCE) is in progress. If an attempt is made to run the ARCHIVE LOG command under these circumstances, error message DSNJ315I or DSNJ316I is issued.

If the system was not fully quiesced (as determined by the number of users which could not be quiesced), error message DSNJ317I is issued and ARCHIVE LOG command processing is terminated. The current active log data set is not truncated and switched to the next available active log data set, and the archive log is not created.

Canceling log offloads: It is possible for the offload of an active log to be suspended when something goes wrong with the offload process, such as a problem with allocation or tape mounting. Issuing ARCHIVE LOG CANCEL OFFLOAD interrupts the offload process and restarts the offload. The command causes an abnormal termination of the offload task, which can result in a dump.

Use ARCHIVE LOG CANCEL OFFLOAD only if the offload task is no longer functioning, or if you want to restart a previous offload attempt that failed.

Demand on DB2 resources: Using the option MODE(QUIESCE) during times of peak activity or during periods in which time is critical causes a significant disruption in the availability of DB2 for all users of DB2 resources.

Interaction with DISPLAY THREAD: The command DISPLAY THREAD issues message DSNV400I, indicating that an ARCHIVE LOG MODE(QUIESCE) command is active.

Quiescing members of a data sharing group: It is not possible to quiesce a single member of a data sharing group. When MODE(QUIESCE) is specified in a data sharing group, the entire group is quiesced.

Executing ARCHIVE LOG while logging is suspended: While logging is suspended by SET LOG SUSPEND, do not use ARCHIVE LOG unless CANCEL OFFLOAD is specified. If logging is suspended, issue SET LOG RESUME to resume logging before issuing ARCHIVE LOG.

Examples

Example 1: Truncate the current active log data sets and initiate an asynchronous job to offload the truncated data sets. No quiesce processing occurs.

```
-ARCHIVE LOG
```

Example 2: Initiate a quiesce period. If all DB2 update activity is stopped within this period, truncate the current active log data set and switch to the next available active log data set. Let the value in the field QUIESCE PERIOD of installation panel DSNTIPA determine the length of the quiesce period. The MODE(QUIESCE) processing is asynchronous.

If the DB2 subsystem can successfully block all update activity before the quiesce period ends, it proceeds to the next processing step. If the quiesce time period is insufficient to successfully quiesce the DB2 subsystem, the active log data sets are not truncated and the archive does not occur.

```
-ARCHIVE LOG MODE(QUIESCE)
```

Example 3: Initiate a quiesce period. If all DB2 update activity is stopped within this period, truncate the current active log data set and switch to the next available active log data set. The maximum length of the quiesce processing period is seven minutes (420 seconds) and the processing is synchronous for the entire seven minutes.

If the DB2 subsystem can successfully block all update activity before the quiesce period ends, it proceeds to the next processing step. If the quiesce time period is insufficient to successfully quiesce the DB2 subsystem, the active log data sets are not truncated and the archive does not occur.

```
-ARCHIVE LOG MODE(QUIESCE) WAIT(YES) TIME(420)
```

Example 4: In a data sharing environment, initiate a quiesce period for all members of the data sharing group. If all DB2 update activity is stopped within this period, truncate the current active log data set and switch to the next available active log data set. Specify a quiesce time period of 10 minutes (600 seconds) to override the value in the field QUIESCE PERIOD of installation panel DSNTIPA for member

DB1G. If the update activity has not quiesced after the 10 minute quiesce period, the command fails and new update activity is allowed to proceed.

```
-DB1G ARCHIVE LOG MODE(QUIESCE) TIME(600)
```

Example 5: In a data sharing environment, truncate the active log data sets for group member DB2G and initiate an asynchronous job to offload the truncated data sets, without any quiesce processing. In this example, SCOPE(MEMBER) is used by default.

```
-DB2G ARCHIVE LOG
```

Example 6: In a data sharing environment, truncate the data sets for all members of the data sharing group and initiate an asynchronous job to offload the truncated data sets, without any quiesce processing.

```
-DB2G ARCHIVE LOG SCOPE(GROUP)
```

Chapter 13. BIND PACKAGE (DSN)

The DSN subcommand BIND PACKAGE builds an application package. DB2 records the description of the package in the catalog tables and saves the prepared package in the directory. For more information on using BIND PACKAGE, see Part 5 of *DB2 Application Programming and SQL Guide*.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 53
- "Option descriptions" on page 54
- "Examples" on page 54

Environment

You can use BIND PACKAGE from DB2I, or from a DSN session under TSO that runs in either the foreground or background.

Data sharing scope: Group

Authorization

The package owner must have authorization to execute **all** statements embedded in the package for BIND PACKAGE to build a package without producing error messages. (The SYSADM authority includes this authorization.) For VALIDATE(BIND), DB2 verifies the authorization at bind time. For VALIDATE(RUN), DB2 verifies the authorization initially at bind time, but if the authorization check fails, DB2 rechecks it at run time.

The required authorization to add a new package or a new version of an existing package depends on the value of field BIND NEW PACKAGE on installation panel DSNTIPP. The default value is BINDADD.

Table 12 on page 50 summarizes the required authorization to run BIND PACKAGE, depending on the bind options that you specify, and in the case of the ADD option, the value of field BIND NEW PACKAGE.

© Copyright IBM Corp. 1983, 2008

Table 12. Summary of privileges needed for BIND PACKAGE options

Bind option	Installation panel field BIND NEW PACKAGE	Authorization required to run BIND PACKAGE
ADD, using the default owner or primary authorization ID	BINDADD	The primary authorization ID (default owner) must have one of the following to add a new package or new version of an existing package to a collection: • The BINDADD system privilege and either the CREATE IN privilege or PACKADM authority on the collection or on all collections • SYSADM or SYSCTRL authority
	BIND	The primary authorization ID (default owner) must have one of the following to add a new package or a new version of an existing package to a collection: • The BINDADD system privilege and either the CREATE IN privilege or PACKADM authority on the collection or on all collections • SYSADM or SYSCTRL authority • PACKADM authority on the collection or on all collections • The BIND package privilege (can only add a new version of an existing package)

Table 12. Summary of privileges needed for BIND PACKAGE options (continued)

Bind option	Installation panel field BIND NEW PACKAGE	Authorization required to run BIND PACKAGE
ADD, specifying an OWNER other than the primary authorization ID (1)	BINDADD	If the binder does not have SYSADM or SYSCTRL authority, the authorization ID of the OWNER must have one of the following to add a new package or new version of an existing package to a collection: • The BINDADD system privilege and either the CREATE IN privilege or PACKADM authority on the collection or on all collections • SYSADM or SYSCTRL authority
	BIND	If the binder does not have SYSADM or SYSCTRL authority, the authorization ID of the OWNER must have one of the following to add a new package or new version of an existing package to a collection: • The BINDADD system privilege and either the CREATE IN privilege or PACKADM authority on the collection or on all collections • SYSADM or SYSCTRL authority • PACKADM authority on the collection or on all collections • The BIND package privilege (can only add a new version of an existing package)
REPLACE, using the default owner or primary authorization ID	BINDADD or BIND	Primary authorization ID must have one of the following: • Ownership of the package • BIND privilege on the package • PACKADM authority on the collection or on all collections • SYSADM or SYSCTRL authority
REPLACE, specifying an OWNER other than the primary authorization ID (1)	BINDADD or BIND	If the binder does not have SYSADM or SYSCTRL authority, the authorization ID of the OWNER must have one of the following: • BIND privilege on the package • PACKADM authority on the collection or on all collections • SYSADM or SYSCTRL authority

Table 12. Summary of privileges needed for BIND PACKAGE options (continued)

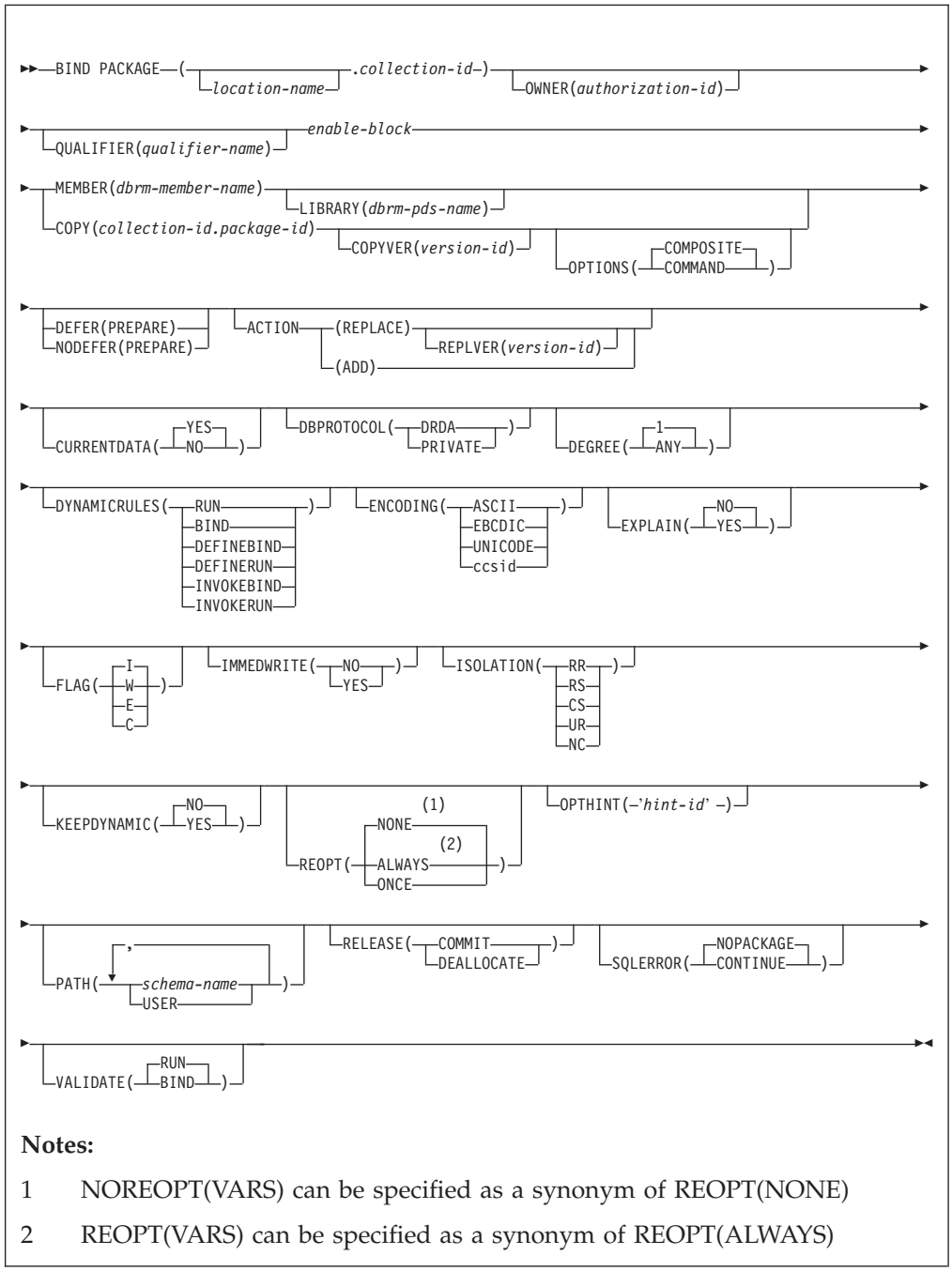
Bind option	Installation panel field BIND NEW PACKAGE	Authorization required to run BIND PACKAGE
COPY	BINDADD or BIND	The primary or secondary authorization ID of the binder or OWNER must have one of the following on the package being copied: • Ownership of the package • COPY privilege on the package • BINDAGENT privilege from the owner of the package • PACKADM authority on the collection or on all collections • SYSADM or SYSCTRL authority

Notes:

1. If any of the authorization IDs of the process has the SYSADM authority or SYSCTRL authority, OWNER *authorization-id* can be any value. If any of the authorization IDs has the BINDAGENT privilege granted from the owner, then *authorization-id* can specify the grantor as OWNER. Otherwise, the OWNER *authorization-id* must be one of the primary or secondary authorization IDs of the binder.

For additional information about the required authorization to execute BIND PACKAGE see Part 5 (Volume 2) of *DB2 Administration Guide*.

Syntax

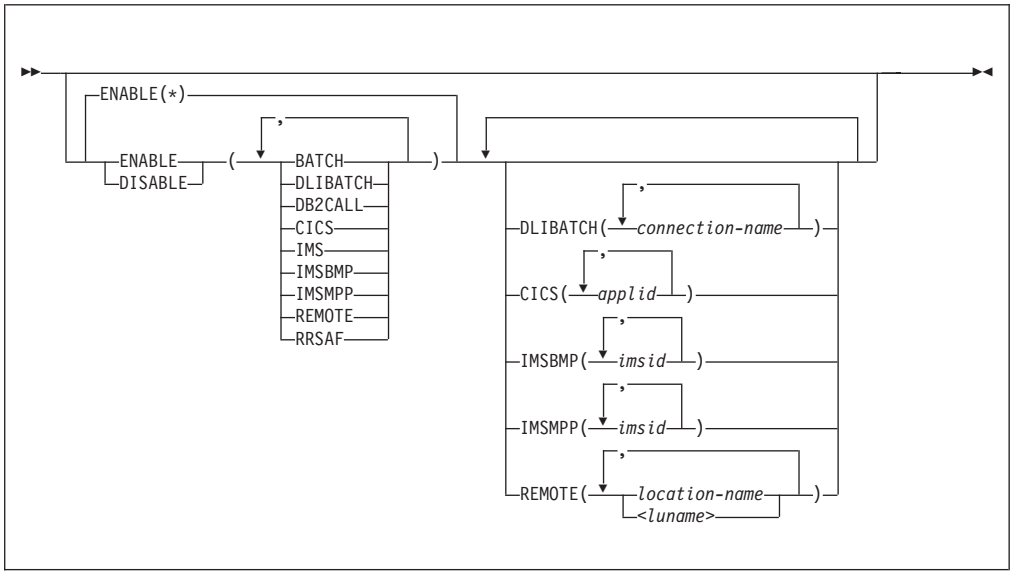


Notes:

1 NOREOPT(VARS) can be specified as a synonym of REOPT(NONE)

2 REOPT(VARS) can be specified as a synonym of REOPT(ALWAYS)

enable-block:



Option descriptions

For descriptions of the options shown in the syntax diagram, see Chapter 15, "BIND and REBIND options," on page 63.

Examples

Example 1: Replace version APRIL_VERSION of package TEST.DSN8BC81 at local location USIBMSTODB22 with another version of the package. The new version (or it could be the same) is in the DBRM DSN8BC81. If the DBRM contains no version ID, the version ID of the package defaults to the empty string. The package runs only from the TSO BATCH environment, and from the CICS environment if the connection ID is CON1. The name PRODUCTN qualifies all unqualified table, view, alias and index names.

```
BIND PACKAGE (USIBMSTODB22.TEST) -
  MEMBER (DSN8BC81) -
  ACTION (REPLACE) REPLVER (APRIL_VERSION) -
  QUALIFIER (PRODUCTN) -
  ENABLE (BATCH, CICS) CICS (CON1)
```

Example 2: UR isolation acquires almost no locks. It is fast and causes little contention, but it reads uncommitted data. Do not use ISOLATION(UR) unless you are sure that your applications and end users can accept the logically inconsistent data that can occur, such as in the case of this example.

Assume that a supervisor routinely executes SQL statements using SPUFI to check the status of parts as they go through the assembly process and to update a table with the results of her inspection. She does not need to know the exact status of the parts; a small margin of error is acceptable.

The supervisor queries the status of the parts from a production table called ASSEMBLY-STATUS and makes the updates in a non-production table called REPORTS. She uses the SPUFI option AUTOCOMMIT NO and has the habit of leaving data on the screen while she performs other tasks.

If the supervisor executes a version of SPUFI that is bound with ISOLATION(UR), the query for the status of the parts executes without acquiring locks using UR isolation level and the update executes using CS isolation level. Thus, the query does not inadvertently hold locks in the production table, which interfers with the production jobs, and the supervisor has data good enough for her purposes.

The SPUFI application is bound as follows:

```
BIND PACKAGE(DSNESPUR) -
  COPY(DSNESPCS.DSNESM68) -
  ACTION(ADD) -
  ISOLATION(UR)
```

Chapter 14. BIND PLAN (DSN)

The DSN subcommand BIND PLAN builds an application plan. All DB2 programs require an application plan to allocate DB2 resources and support SQL requests made at run time. For more information on using BIND PLAN, see Part 5 of *DB2 Application Programming and SQL Guide*.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 59
- "Option descriptions" on page 60
- "Examples" on page 60

Environment

You can use BIND PLAN through DB2I, or from a DSN session under TSO that runs in either the foreground or background.

Data sharing scope: Group

Authorization

The plan owner must have authorization to execute **all** SQL statements embedded in the plan for BIND PLAN to build a plan without producing error messages. This excludes statements included in DBRMs that are bound to packages included in the package list of the plan. The SYSADM authority includes this authorization. For VALIDATE(BIND), DB2 verifies the authorization at bind time. For VALIDATE(RUN), DB2 verifies the authorization initially at bind time, but if the authorization check fails, DB2 rechecks it at run time.

Table 13 explains the authorization required to run BIND PLAN, depending on the options specified.

Table 13. Summary of privileges needed for BIND PLAN options

Option	Authorization required to run BIND PLAN
ADD, using the default owner or primary authorization ID	Primary authorization ID (default owner) must have one of the following: • BINDADD privilege • SYSADM or SYSCTRL authority
ADD, specifying an OWNER other than the primary authorization ID	If the binder does not have SYSADM or SYSCTRL authority, the authorization ID of the new OWNER must have one of the following: • BINDADD privilege • SYSADM or SYSCTRL authority
REPLACE, using the default owner or primary authorization ID	Primary authorization ID of the process must have one of the following: • Ownership of the plan • BIND privilege on the plan • SYSADM or SYSCTRL authority

© Copyright IBM Corp. 1983, 2008

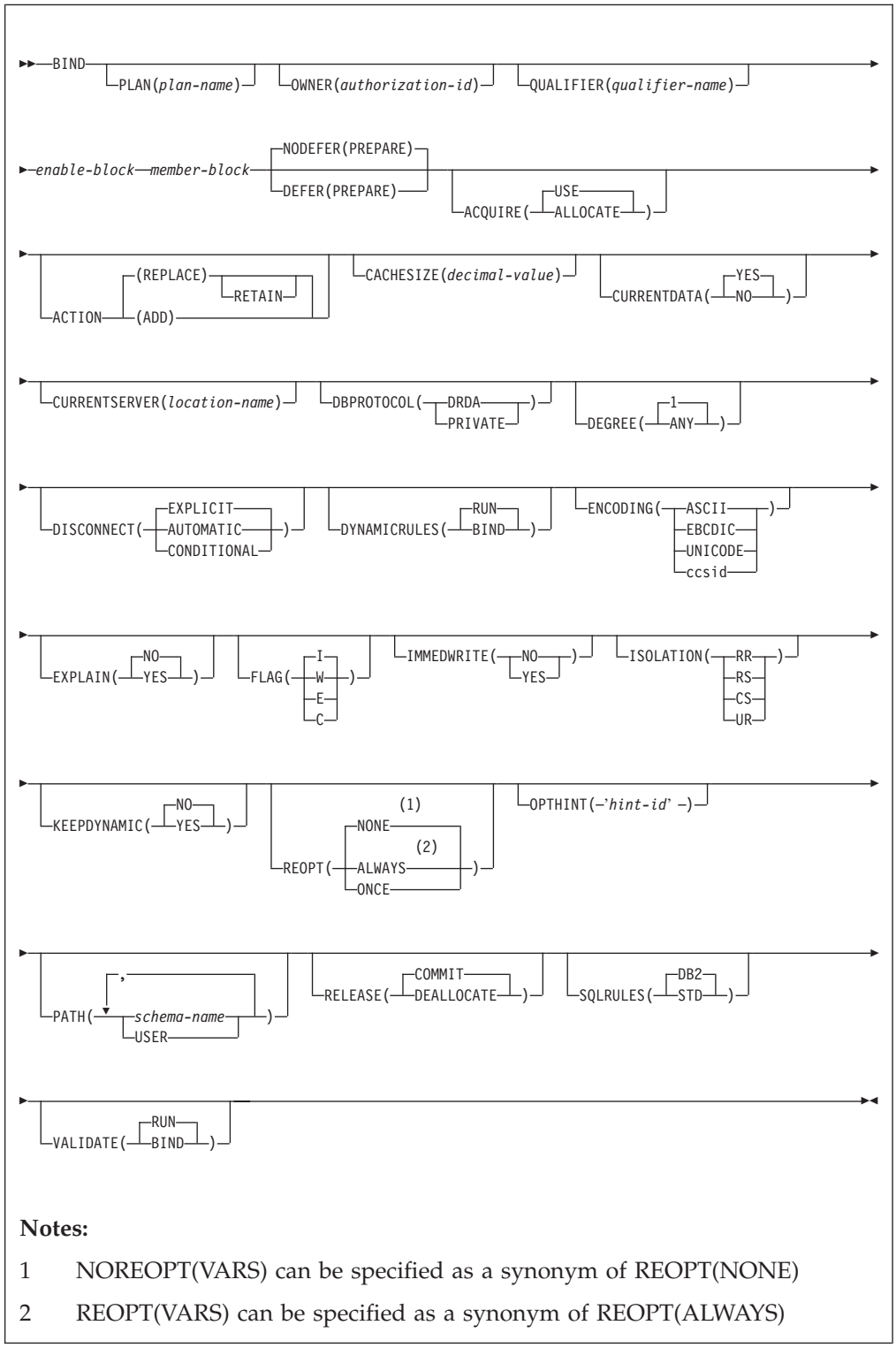
Table 13. Summary of privileges needed for BIND PLAN options (continued)

Option	Authorization required to run BIND PLAN
REPLACE, specifying an OWNER other than the primary authorization ID	If the binder does not have SYSADM or SYSCTRL authority, the authorization ID of the OWNER must have one of the following: • Ownership of the plan • BIND privilege on the plan • SYSADM or SYSCTRL authority
PKLIST, specifying individual packages	Authorization ID of the process must include one of the following: • EXECUTE authority on each package specified in the PKLIST • PACKADM authority on specific collections that contain the packages or on all collections • SYSADM authority
PKLIST, specifying (*), indicating all packages in the collection	Authorization ID of the process must include one of the following: • EXECUTE authority on *collection-id*.* • PACKADM authority on specific collections that contain the packages or on all collections • SYSADM authority

Specifying the OWNER for ADD and REPLACE: If any of the authorization IDs of the process has SYSADM authority or SYSCTRL authority, OWNER *authorization-id* can be any value. If any of the authorization IDs has the BINDAGENT privilege granted from the owner, *authorization-id* can specify the grantor as OWNER. Otherwise, OWNER *authorization-id* must be one of the primary or secondary authorization IDs of the binder.

For additional information about the required authorization to execute BIND PLAN, see Part 5 of *DB2 Application Programming and SQL Guide*.

Syntax

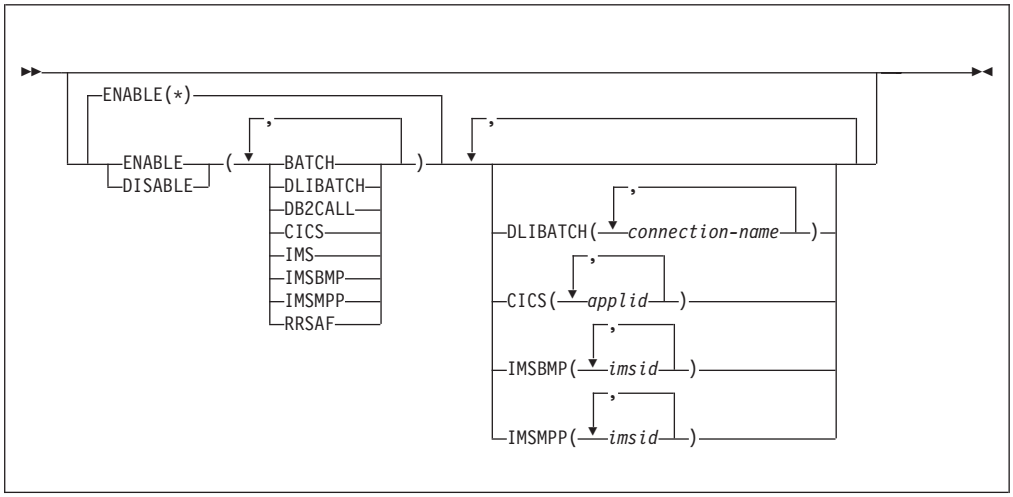


Notes:

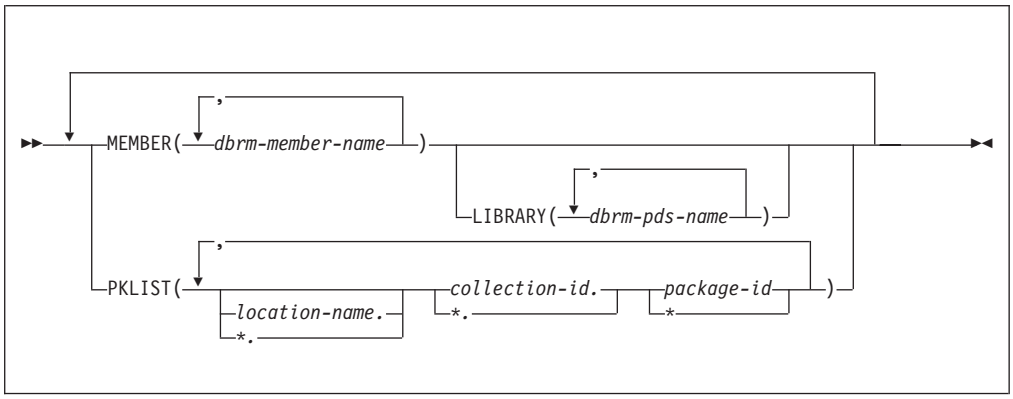
1 NOREOPT(VARS) can be specified as a synonym of REOPT(NONE)

2 REOPT(VARS) can be specified as a synonym of REOPT(ALWAYS)

enable-block:



member-block:



Option descriptions

For descriptions of the options shown in the syntax diagram, see Chapter 15, "BIND and REBIND options," on page 63.

Examples

Example 1: This subcommand creates a new plan called IMSONLY. The SQL statements for the plan are in the DBRM member DSN8BC81. An ISOLATION level of cursor stability (CS) provides maximum concurrency when you run the plan, and protects database values only while the program uses them. DEPTM92 owns the plan, but PRODUCTN qualifies any unqualified table, view, index, and alias names that are referenced in the DBRM.

A cache size of 0 indicates that users will not run the plan repeatedly. Caching the names of users authorized to run the plan helps only when the same user runs the plan repeatedly while it is in the EDM pool. Because this is not the case with this plan, there is no need to reserve space in the EDM pool for a cache that the plan does not use.

The option ENABLE(IMS) runs the plan only from an IMS environment (DLI Batch, BMP and MPP). If you attempt to run the plan from another environment, such as TSO Batch, the plan allocation fails.

```
BIND PLAN(IMSONLY) -
  MEMBER(DSN8BC81) -
  ACTION(ADD) -
  ISOLATION(CS) -
  OWNER(DEPTM92) -
  QUALIFIER(PRODUCTN) -
  CACHESIZE -
  ENABLE(IMS)
```

Example 2: If the DBRM of plan IMSONLY in Example 1 contains both embedded and dynamic SQL statements and you want to allow other users to run the plan, you must grant the EXECUTE privilege on plan IMSONLY to those users' authorization IDs. However, because the EXECUTE privilege on a plan is sufficient authority to run embedded SQL statements in a DBRM but is not sufficient authority to run dynamic SQL statements, you must also do one of the following:

- Use the SQL GRANT statement to grant the necessary privileges on the objects (tables, views, aliases, and indexes) referenced in the dynamic SQL statements to the users' authorization IDs, or
- BIND the plan IMSONLY with the option DYNAMICRULES(BIND) as follows:

  ```
  BIND PLAN(IMSONLY) -
    MEMBER(DSN8BC81) -
    ACTION(ADD) -
    ISOLATION(CS) -
    OWNER(DEPTM92) -
    QUALIFIER(PRODUCTN) -
    CACHESIZE(0) -
    ENABLE(IMS) -
    DYNAMICRULES(BIND)
  ```

To allow other users having only the EXECUTE privilege on a plan to run both the embedded and dynamic SQL statements, you must bind that plan with the option DYNAMICRULES(BIND). When DYNAMICRULES(BIND) is in effect for plan IMSONLY:

- A single authorization ID, the authorization ID for DEPTM92, is used for authorization checking of both the embedded and dynamic SQL statements in the DBRM.
- PRODUCTN is the implicit qualifier of unqualified object names referenced in both the embedded and dynamic SQL statements in the DBRM.

Example 3: This subcommand creates a new plan called CICSONLY. The plan specifies an ISOLATION level of cursor stability (CS). DEPTM12 owns the plan, but TESTSYS qualifies any unqualified table, view, index, and alias names referenced in the DBRM. A cache size of 0 indicates that users will not run the plan repeatedly.

The option ENABLE(CICS) CICS(CON1) runs the plan only from CICS VTAM® node CON1 which is specified in the APPLID parameter of the CICS SIT table. If you attempt to run the plan from another environment or from another CICS VTAM note, the run attempt fails.

```
BIND PLAN(CICSONLY) -
  MEMBER(DSN8BC81) -
  ACTION(ADD) -
  ISOLATION(CS) -
```

```
                        OWNER(DEPTM12) -
                        QUALIFIER(TESTSYS) -
                        CACHESIZE(0) -
                        ENABLE(CICS) CICS(CON1)
```

Chapter 15. BIND and REBIND options

This chapter lists the options you can use for binding or rebinding plans and packages. Some of the options are common for both bind and rebind and for both plans and packages.

Defaults: The *default* for an option is the value used if you omit the entire option.

A default of *plan value* for BIND PACKAGE means that the default is the same as the value determined during the bind or rebind of the plan to which the package is appended at run time.

A default of *existing value* for REBIND PLAN or REBIND PACKAGE means that the default is the value that was determined during the previous bind or rebind of the plan or package that you are rebinding.

For all other cases, the option descriptions note the specific defaults, which DB2 assigns at bind time. If a specific default value exists, that value is underlined.

Catalog records: The DB2 catalog records information about plans and packages, chiefly in the tables SYSIBM.SYSPLAN and SYSIBM.SYSPACKAGE. The descriptions of where the options record information omit the constant qualifier, SYSIBM, of those table names.

ACQUIRE (**USE**) (**ALLOCATE**) **On: BIND and REBIND PLAN**

Determines whether to acquire resources for DBRMs specified in the MEMBER list when the application first accesses them or when the plan is allocated. Local or remote packages associated with the plan acquire their resources when the application first accesses them.

(USE)
Acquires table space locks only when the application program bound to the plan first uses them.

(ALLOCATE)
Acquires all table space locks when the plan is allocated. The value has no effect on dynamic SQL statements, which always use ACQUIRE(USE).

If you use ACQUIRE(ALLOCATE), you must also use RELEASE(DEALLOCATE). ACQUIRE(ALLOCATE) can increase the plan size, because additional items become resident in the plan.

Defaults:

Process	Default value
BIND PLAN	USE
BIND PACKAGE	N/A
REBIND PLAN	Existing value
REBIND PACKAGE	N/A

© Copyright IBM Corp. 1983, 2008

There is no ACQUIRE option for packages. A package always acquires resources when it first uses them, as if you specified ACQUIRE(USE). See Part 5 of *DB2 Application Programming and SQL Guide*.

Catalog record: Column ACQUIRE of table SYSPLAN.

For more information about:

- How the option affects locking and concurrency, see Part 5 of *DB2 Application Programming and SQL Guide* or Part 4 of *DB2 Application Programming and SQL Guide* .
- How the option improves the performance of selective partition locking, see Part 5 of *DB2 Application Programming and SQL Guide* or Part 4 of *DB2 Application Programming and SQL Guide*.
- Estimating the size of a plan, see Part 2 of *DB2 Administration Guide*.

ACTION	(**REPLACE**) (**REPLACE**) **RPLVER** (BIND PACKAGE only) (**REPLACE**) **RETAIN** (BIND PLAN only) (**ADD**)	**On: BIND PLAN and PACKAGE**

Determines whether the object (plan or package) replaces an existing object with the same name or is new.

(REPLACE)
The object replaces an existing one with the same identifier, and a new entry replaces the old one in the catalog table SYSPLAN or SYSPACKAGE. If no object with the given identifier already exists, the bind process creates the new object and a new entry.

The authorization ID designated explicitly or implicitly by the option OWNER becomes the owner of the new object. If that authorization ID is not the previous owner, all grants of privileges for the object that the previous owner issued change to name the new owner as the grantor.

If the bind fails, the old object and its entry remain.

For BIND PACKAGE: You cannot use REPLACE with a remote package bound with either of the options ENABLE or DISABLE. The attempt causes the bind to fail.

REPLVER(*version-id*) (For BIND PACKAGE only)
Replaces a specific version of the package, identified by *version-id*. If the package with the specified *version-id* does not exist, the bind fails.

The default for *version-id* comes from the DBRM if you use the MEMBER option on BIND, or from the COPYVER option if you use the COPY option.

RETAIN (For BIND PLAN only)
Preserves EXECUTE privileges when you replace the plan. If ownership of the plan changes, the new owner grants the privileges BIND and EXECUTE to the previous owner.

RETAIN is *not* the default. If you do not specify RETAIN, everyone but the plan owner loses the EXECUTE privilege (but not the BIND privilege). If plan ownership changes, the new owner grants the BIND privilege to the previous owner.

(ADD)
Adds a new object, but does not replace an existing one. If the object name already exists in the catalog, the bind fails. If the bind fails for any reason, the bind process does not produce a new package or plan and makes no entry in the catalog.

Replacing a version of a package (REPLVER): This section describes the effect of ACTION(REPLACE) REPLVER in four situations. Here, DBRM1 is the member name and A and B represent the names of two versions of the package. Suppose you bind version A with this command:

```
BIND PACKAGE(COLL1) MEMBER(DBRM1) ACTION(REPLACE) REPLVER(B)
```

- If neither DBRM1, version A, nor version B exist in the DB2 catalog, the command fails because version B is not in the catalog. No new package is added.
- If DBRM1 and version B, but not version A, exist in the DB2 catalog, then version A replaces version B. As a result, version A exists in the catalog, and version B no longer exists in the catalog.
- If DBRM1 and version A exist in the catalog, but not version B, the command fails because version B is not in the catalog. Version A continues to exist.
- If DBRM1 and both versions A and B exist in the catalog, the command fails because version A already exists.

Defaults:

Process	Default value
BIND PLAN	REPLACE
BIND PACKAGE	REPLACE
REBIND PLAN	N/A
REBIND PACKAGE	N/A

Catalog record: Tables SYSPLAN or SYSPACKAGE.

CACHESIZE (*value of field PLAN AUTH CACHE*) (*decimal-value*) **On: BIND and REBIND PLAN**

Determines the size (in bytes) of the authorization cache acquired in the EDM pool for the plan. At run time, the authorization cache stores user IDs authorized to run. Consulting the cache can avoid a catalog lookup for checking authorization to run the plan.

decimal-value
The size of the cache can range from 0 to 4096. Nonzero values that are not multiples of 256 round to the next highest multiple of 256. CACHESIZE(0) specifies creating no cache when the plan runs.

Defaults:

Process	Default value
BIND PLAN	*value of field PLAN AUTH CACHE* on installation panel DSNTIPP, which has a default of 0
BIND PACKAGE	N/A
REBIND PLAN	Existing value

REBIND PACKAGE	N/A

Catalog record: Column CACHESIZE of table SYSPLAN.

For additional information on determining an optimal cache size, see Part 5 of *DB2 Application Programming and SQL Guide*.

COPY (*collection-id.package-id*) **On: BIND PACKAGE**
(*collection-id.package-id*) **COPYVER**

Determines that you are copying an existing package and names that package. Copying the package recalculates the access paths in the copy.

To create a remote copy, this option copies SQL statements from a package at your local server. Therefore, you must hold the COPY privilege or its equivalent at the **local** server.

collection-id
The name of the collection that contains the package to copy, as listed in column COLLID of catalog table SYSPACKAGE.

package-id
The name of the package to copy, as listed in column NAME of catalog table SYSPACKAGE.

COPYVER(*version-id***)**
Determines the version of the package to copy. The default for *version-id* is the empty string.

Restrictions:

- *collection-id.package-id* must identify a package on the local server.
- You cannot copy to a package in the same collection. If you make the copy on the local server, *collection-id.* on the COPY option must not name the collection used on the PACKAGE option.
- DB2 uses the ISO format for output values unless the SQL statement explicitly specifies a different format. Input values can be specified in one of the standard formats, or in a format that is recognized by the server's local date/time exit.

Defaults:

Process	Default value
BIND PLAN	N/A
BIND PACKAGE	None
REBIND PLAN	N/A
REBIND PACKAGE	N/A

COPY has **no default**. If you do not use COPY, you must use MEMBER. You cannot use both options.

The option values of the package copied (**except** the values of ENABLE, DISABLE, OWNER, and QUALIFIER) become the defaults for binding the new package. You can override a default by choosing a new value for an option on the BIND PACKAGE command.

Copy packages to remote servers: To copy and bind packages from DB2 UDB for z/OS Version 8 to some other server that does not support all the new BIND options in Version 8, use the new OPTIONS(COMMAND) option on BIND PACKAGE COPY. Any options you do not explicitly specify on the BIND PACKAGE subcommand are set to the server's defaults. Using this option can prevent bind errors when you bind and copy packages to servers other than DB2 UDB for z/OS Version 8.

Catalog record: Column COPY of table SYSPACKAGE.

CURRENTDATA	(**YES**) (**NO**)	**On: BIND and REBIND PLAN and PACKAGE, REBIND TRIGGER PACKAGE**

Determines whether to require data currency for read-only and ambiguous cursors when the isolation level of cursor stability is in effect. It also determines whether block fetching can be used for distributed, ambiguous cursors.

For more information about updating the current row of a cursor, block fetching, and data currency, see Part 4 of *DB2 Application Programming and SQL Guide*.

(YES) Specifies that currency is required for read-only and ambiguous cursors. DB2 acquires page or row locks to ensure data currency. Block fetching for distributed, ambiguous cursors is inhibited.

(NO) Specifies that currency is not required for read-only and ambiguous cursors. Block fetching for distributed, ambiguous cursors is allowed.

If your application attempts to dynamically prepare and execute a DELETE WHERE CURRENT OF statement against an ambiguous cursor, after that cursor is opened, use of CURRENTDATA(NO) is not recommended. You receive a negative SQLCODE if your application attempts a DELETE WHERE CURRENT OF statement for any of the following cursors:

- A cursor that is using block fetching
- A cursor that is using query parallelism
- A cursor that is positioned on a row that is modified by this or another application process

Restriction for remote rebinds: You cannot use CURRENTDATA when rebinding a package at a remote server. To change the value of CURRENTDATA, you can:
- Issue BIND REPLACE, remotely or locally.
- Free the package and issue BIND ADD, remotely or locally.
- Rebind the package locally at the location where the package resides.

Defaults:

Process	Default value
BIND PLAN	YES
BIND PACKAGE	YES
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

Catalog record: Column DEFERPREP of table SYSPACKAGE and column EXPREDICATE of table SYSPLAN.

CURRENTSERVER (*location-name*) **On: BIND and REBIND PLAN**

Determines the location to connect to before running the plan. The column CURRENTSERVER in catalog table SYSPLAN records the value of *location-name*. The special register CURRENT SERVER also receives that value at the server when the plan is allocated. When the plan runs, the requester implicitly uses a type 1 CONNECT statement to that location.

You can use CURRENTSERVER to cause a local application to use data from a remote server without changing the application; however, using CURRENTSERVER causes poor performance and should be avoided where possible. Avoid using CURRENTSERVER with applications that contain explicit CONNECT statements. The implicit type 1 CONNECT statement that is used by CURRENTSERVER causes any explicit CONNECT statement issued in the application to be type 1, even if the application was precompiled with the default type 2.

location-name
The name of the location to connect to. The catalog table SYSIBM.LOCATIONS must contain this name. If the table does not exist, if the table does not contain the DBMS, or if there are no packages at that location, warning messages occur.

SQL return codes: CURRENTSERVER causes DB2 to execute a type 1 CONNECT statement. DB2 does not display or report to the application program any warnings that this CONNECT returns. To display the warnings, use explicit CONNECT statements rather than the CURRENTSERVER bind option.

Defaults:

Process	Default value
BIND PLAN	Local DBMS (regardless of the name of the local location)
BIND PACKAGE	N/A
REBIND PLAN	Existing value
REBIND PACKAGE	N/A

Catalog record: Column CURRENTSERVER of table SYSPLAN.

DBPROTOCOL (**DRDA**) (**PRIVATE**) **On: BIND and REBIND PLAN and PACKAGE**

Specifies which protocol to use when connecting to a remote site that is identified by a three-part name statement.

For DRDA®, a package must be bound to each remote site that is referenced by a three-part name statement. Specify DRDA to inform DB2 that the three-part name statements in the plan or package are to be converted to DRDA protocol.

If you specify an option on the BIND PACKAGE command, DB2 uses that remote access method for the package statements, regardless of the BIND PLAN option. For remote bind, the default is the system default at the remote site.

If you specify an option on the BIND PLAN statement, that information is stored in table SYSPLAN.

(DRDA)
DBPROTOCOL DRDA is passed on BIND PACKAGE, BIND PLAN, REBIND PACKAGE, or REBIND PLAN invocation.

(PRIVATE)
DBPROTOCOL PRIVATE is passed on BIND PACKAGE, BIND PLAN, REBIND PACKAGE, or REBIND PLAN invocation. An application that uses DB2 private protocol access cannot include SQL statements that were added to DB2 after Version 7.

Defaults:

Process	Default value
BIND PLAN	DRDA
BIND PACKAGE	System default
REBIND PLAN	Value that was specified the last time that the plan was bound
REBIND PACKAGE	Value that was specified the last time that the package was bound

Catalog record: Column DBPROTOCOL of tables SYSPACKAGE and SYSPLAN.

NODEFER(PREPARE)
DEFER(PREPARE)

On: BIND and REBIND PLAN and PACKAGE

Determines whether to defer preparation for dynamic SQL statements that refer to remote objects, or to prepare them immediately. If you defer preparation, the dynamic statement prepares when DB2 first encounters a statement of the type EXECUTE, OPEN, or DESCRIBE that refers to the dynamic statement.

For BIND and REBIND PACKAGE, if neither option is specified, and REOPT(NONE) applies:

- For local bind the package inherits the plan's option at run time.
- For remote bind the default is NODEFER(PREPARE) at the remote DB2 server.

If neither DEFER nor NODEFER is specified and REOPT(ALWAYS) applies, DEFER(PREPARE) is the default value.

You cannot use both DEFER(PREPARE) and NODEFER(PREPARE).

NODEFER(PREPARE)
Does not defer preparation.

DEFER(PREPARE)
Defers preparation.

DEFER(PREPARE) and distributed processing: To improve performance, consider using DEFER(PREPARE) when binding dynamic or static SQL for DB2 private protocol access and when binding dynamic SQL for DRDA access. Specify the bind option DEFER(PREPARE) instead of NODEFER(PREPARE). DB2 does not prepare the dynamic SQL statement until that statement executes. This reduces network traffic, which improves the performance of the dynamic SQL statement.

To defer the preparation of an SQL statement in an application, bind or rebind the application with the option DEFER(PREPARE). This defers PREPARE messages for SQL statements that refer to a remote object until either:
- The statement executes
- The application requests a description of the results of the statement

If you choose to defer PREPARE statements, after the EXECUTE or DESCRIBE statement, you should code your application to handle any SQL error codes or SQLSTATEs that the PREPARE statement might return. You can defer PREPARE statements only if you specify the bind option DEFER(PREPARE).

Defaults:

Process	Default value
BIND PLAN	NODEFER
BIND PACKAGE	Plan value
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

Catalog record: Column DEFERPREP of table SYSPLAN and column DEFERPREPARE of table SYSPACKAGE.

DEGREE (**1**) (**ANY**) — **On: BIND and REBIND PLAN and PACKAGE**

Determines whether to attempt to run a query using parallel processing to maximize performance.

For plans, the value of DEGREE applies only to the DBRMs bound directly to the plan (named in the MEMBER option on BIND PLAN), and has no affect on PKLIST names. The value has no effect on dynamic SQL statements, which use the value of the special register CURRENT DEGREE. The value of the special register can be changed by executing the SET CURRENT DEGREE statement.
(1) Prohibits parallel processing.
(ANY) Allows parallel processing.

Limitations: If you bind plans or packages using DEGREE=ANY, the space required in the EDM pool could increase by 50%–70%.

Defaults:

Process	Default value
BIND PLAN	1
BIND PACKAGE	1
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

Catalog record: Column DEGREE of tables SYSPACKAGE and SYSPLAN.

DISCONNECT (**EXPLICIT**) (**AUTOMATIC**) (**CONDITIONAL**) — **On: BIND and REBIND PLAN**

Determines which remote connections to destroy during commit operations. The option applies to any application process that uses the plan and has remote connections of any type. Regardless of the value of this option, a commit operation destroys all connections in the release pending state. You can put a connection in the release pending state using the SQL statement RELEASE.

(EXPLICIT)
Destroy only connections in the release pending state. This value allows you maximum flexibility for controlling remote connections.

(AUTOMATIC)
Destroy all remote connections.

(CONDITIONAL)
Destroy all remote connections unless an open cursor defined as WITH HOLD is associated with the connection.

Defaults:

Process	Default value
BIND PLAN	EXPLICIT
BIND PACKAGE	N/A
REBIND PLAN	Existing value
REBIND PACKAGE	N/A

Catalog record: Column DISCONNECT of table SYSPLAN.

DYNAMICRULES

(**RUN**)
(**BIND**)
(**DEFINEBIND**) (BIND and REBIND PACKAGE only)
(**DEFINERUN**) (BIND and REBIND PACKAGE only)
(**INVOKEBIND**) (BIND and REBIND PACKAGE only)
(**INVOKERUN**) (BIND and REBIND PACKAGE only)

On: BIND and REBIND PLAN and PACKAGE

Determines what values apply at run time for the following dynamic SQL attributes:
- The authorization ID that is used to check authorization
- The qualifier that is used for unqualified objects
- The source for application programming options that DB2 uses to parse and semantically verify dynamic SQL statements
- Whether dynamic SQL statements can include GRANT, REVOKE, ALTER, CREATE, DROP, and RENAME statements

In addition to the DYNAMICRULES value, the run-time environment of a package controls how dynamic SQL statements behave at run time. The two possible run-time environments are:
- The package runs as part of a stand-alone program
- The package runs as a stored procedure or user-defined function package, or runs under a stored procedure or user-defined function

The combination of the DYNAMICRULES value and the run-time environment determine the values for the dynamic SQL attributes. That set of attribute values is called the dynamic SQL statement *behavior*. The four behaviors are:
- Run behavior
- Bind behavior

- Define behavior
- Invoke behavior

The following DYNAMICRULES option descriptions include a description of the dynamic SQL statement behavior for each run-time environment. This information is summarized in Table 14 on page 74.

(RUN) Processes dynamic SQL statements using the standard attribute values for dynamic SQL statements, which are collectively called *run behavior*:

- DB2 uses the authorization ID of the application process and the SQL authorization ID (the value of the CURRENT SQLID special register) for authorization checking of dynamic SQL statements.
- DB2 uses the authorization ID of the application process and the SQL authorization ID (the value of the CURRENT SQLID special register) as the implicit qualifier of table, view, index, and alias names.
- Dynamic SQL statements use the values of application programming options that were specified during installation. The installation option USE FOR DYNAMICRULES has no effect.
- GRANT, REVOKE, CREATE, ALTER, DROP, and RENAME statements can be executed dynamically.

(BIND)

Processes dynamic SQL statements using the following attribute values, which are collectively called *bind behavior*:

- DB2 uses the authorization ID of the plan or package for authorization checking of dynamic SQL statements.
- Unqualified table, view, index, and alias names in dynamic SQL statements are implicitly qualified with value of the bind option QUALIFIER; if you do not specify QUALIFIER, DB2 uses the authorization ID of the plan or package owner as the implicit qualifier.
- The attribute values that are described in "Common attribute values for bind, define, and invoke behaviors" on page 73.

The values of the authorization ID and the qualifier for unqualified objects are the same as those that are used for embedded or static SQL statements.

(DEFINEBIND)

Processes dynamic SQL statements using one of two behaviors, *define behavior* or *bind behavior*.

When the package is run as or runs under a stored procedure or user-defined function package, DB2 processes dynamic SQL statements using define behavior, which consists of the following attribute values:

- DB2 uses the authorization ID of the user-defined function or stored procedure owner for authorization checking of dynamic SQL statements in the application package.
- The default qualifier for unqualified objects is the user-defined function or stored procedure owner.
- The attribute values that are described in "Common attribute values for bind, define, and invoke behaviors" on page 73.

When the package is run as a stand-alone program, DB2 processes dynamic SQL statements using bind behavior, which is described in 72.

(DEFINERUN)

Processes dynamic SQL statements using one of two behaviors, *define behavior* or *run behavior*.

When the package is run as or runs under a stored procedure or user-defined function package, dynamic SQL statements have define behavior, which is described in 72.

When the package is run as a stand-alone program, DB2 processes dynamic SQL statements using run behavior, which is described in 72.

(INVOKEBIND)
Processes dynamic SQL statements using one of two behaviors, *invoke behavior* or *bind behavior*.

When the package is run as or runs under a stored procedure or user-defined function package, DB2 processes dynamic SQL statements using invoke behavior, which consists of the following attribute values:

- DB2 uses the authorization ID of the user-defined function or stored procedure invoker for authorization checking of dynamic SQL statements in the application package.

  If the invoker is the primary authorization ID of the process or the CURRENT SQLID value, secondary authorization IDs are also checked if they are needed for the required authorization. Otherwise, only one ID, the ID of the invoker, is checked for the required authorization.
- The default qualifier for unqualified objects is the user-defined function or stored procedure invoker.
- The attribute values that are described in "Common attribute values for bind, define, and invoke behaviors."

When the package is run as a stand-alone program, DB2 processes dynamic SQL statements using bind behavior, which is described in 72.

(INVOKERUN)
Processes dynamic SQL statements using one of two behaviors, *invoke behavior* or *run behavior*.

When the package is run as or runs under a stored procedure or user-defined function package, DB2 processes dynamic SQL statements using invoke behavior, which is described in 73.

When the package is run as a stand-alone program, DB2 processes dynamic SQL statements using run behavior, which is described in 72.

Common attribute values for bind, define, and invoke behavior: The following attribute values apply to dynamic SQL statements in plans or packages that have bind, define, or invoke behavior:

- You can execute the statement SET CURRENT SQLID in a package or plan that is bound with any DYNAMICRULES value. However, DB2 does not use the value of CURRENT SQLID as the authorization ID for dynamic SQL statements.

  DB2 always uses the value of CURRENT SQLID as the qualifier for the EXPLAIN output PLAN_TABLE. (If the value of CURRENT SQLID has an alias on PLAN_TABLE and has the appropriate privileges, that PLAN_TABLE is populated.)
- If the value of installation option USE FOR DYNAMICRULES is YES, DB2 uses the application programming default values that were specified during installation to parse and semantically verify dynamic SQL statements. If the value of USE for DYNAMICRULES is NO, DB2 uses the precompiler options to parse and semantically verify dynamic SQL statements. For a list of the application programming defaults that the USE FOR DYNAMICRULES option affects, see Part 5 of *DB2 Application Programming and SQL Guide*.

- GRANT, REVOKE, CREATE, ALTER, DROP, and RENAME statements cannot be executed dynamically.

Remote DB2 servers: For a package that uses DRDA access, DB2 sends the DYNAMICRULES option to the DB2 server at bind time.

For a plan or package that uses DB2 private protocol access, DB2 sends a DYNAMICRULES value of BIND or RUN to the server at run time, using the following rules:

- If the DYNAMICRULES value with which the package is bound is BIND, DEFINEBIND, or INVOKEBIND, DB2 sends a value of BIND to the server.
- If the DYNAMICRULES value with which the package is bound is RUN, DEFINERUN, or INVOKERUN, DB2 sends a value of RUN to the server.

Table 14 summarizes the dynamic SQL statement attribute values for each behavior. For more information about the dynamic SQL attributes that are affected by the DYNAMICRULES option, see Part 5 of *DB2 Application Programming and SQL Guide*.

Table 14. Definitions of dynamic SQL statement behaviors

Dynamic SQL attribute	Value for bind behavior	Value for run behavior	Value for define behavior	Value for invoke behavior
Authorization ID	Package OWNER	Current SQLID	User-defined function or stored procedure owner	Authorization ID of invoker
Default qualifier for unqualified objects	Bind OWNER or QUALIFIER value	Current SQLID	User-defined function or stored procedure owner	Authorization ID of invoker
CURRENT SQLID	Initialized to primary authid. SET SQLID is allowed.	Initialized to primary authid. SET SQLID is allowed.	Initialized to primary authid. SET SQLID is allowed.	Initialized to primary authid. SET SQLID is allowed.
Source for application programming options	As determined by the DSNHDECP parameter DYNRULS	Installation panel DSNHDECP application defaults	As determined by the DSNHDECP parameter DYNRULS	As determined by the DSNHDECP parameter DYNRULS
Can execute GRANT, REVOKE, CREATE, ALTER, DROP, RENAME?	No	Yes	No	No

Defaults:

Process	Default value
BIND PLAN	RUN
BIND PACKAGE	Plan value
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

The default for a package on a remote server is RUN.

Catalog record: Column DYNAMICRULES of tables SYSPACKAGE and SYSPLAN.

ENABLE **DISABLE**	(*) (**BATCH**) (**CICS**) (**CICS**) **CICS**(*applid*, ...) (**DB2CALL**) (**DLIBATCH**) (**DLIBATCH**) **DLIBATCH**(*connection-name*, ...) (**IMS**) (**IMSBMP**) (**IMSBMP**) **IMSBMP**(*imsid*, ...) (**IMSMPP**) (**IMSMPP**) **IMSMPP**(*imsid*, ...) (**REMOTE**) (BIND and REBIND PACKAGE only) (**REMOTE**) **REMOTE** (*location-name*,..., < *luname*>,...) (**RRSAF**)	**On: BIND and REBIND PLAN and PACKAGE**

Determines which connections can use the plan or package. You cannot use both DISABLE and ENABLE. For packages, DISABLE and ENABLE are valid only for local bind operations.

ENABLE
Lists the system connection types that can use the plan or package. Connection types not listed cannot use it.

DISABLE
Lists the system connection types that cannot use the plan or package. Connection types not listed can use it.

With some connection types you can list connection IDs to identify specific connections of the type to disable or enable.
If you list connection IDs as disabled, any connections not listed for the same connection type are enabled.
If you list connection IDs as enabled, any connections not listed for the same connection type are disabled.

A connection ID is valid only after the keyword that names its corresponding connection type.

Connection types:

(*) Specifies all valid connection types. Use only with ENABLE.

(BATCH)
Indicates that all TSO connections are either enabled or disabled for the plan or package.

(CICS)
Identifies the CICS Connection®. All CICS VTAM node names specified in the CICS SIT table are either enabled or disabled for the plan or package.

(CICS) CICS(*applid*, ...)
Identifies the CICS VTAM node name specified in the APPLID parameter of the CICS SIT table. The CICS VTAM node identified by *applid* is either enabled or disabled for the plan or package.

(DB2CALL)
Indicates that the call attachment facility (CAF) connection is either enabled or disabled for the plan or package.

(DLIBATCH)
Identifies the Data Language I (DL/I) Batch Support Facility connection. All connection identifiers from the DDITV02 data set or the job name in the JCL that the DL/I batch support system needs to have are either enabled or disabled for the plan or package.

(DLIBATCH) DLIBATCH(*connection-name***, ...)**
Specifies the connection identifier as from the DDITV02 data set or the job name in the JCL that the DL/I batch support system needs to have. The DL/I batch connection identified by *connection-name* is either enabled or disabled for the plan or package.

(IMS)
Specifies that all Information Management System (IMS) connections, DLIBATCH, IMSBMP, and IMSMPP are either enabled or disabled for the plan or package.

(IMSBMP)
Specifies the IMS connection for the Batch Message Program (BMP) region. All IMS BMP connections identified by the value of IMSID on the CTL parameter EXEC are either enabled or disabled for the plan or package.

(IMSBMP) IMSBMP(*imsid***, ...)**
Specifies the value of IMSID on the CTL parameter EXEC. The IMS BMP connection identified by *imsid* is either enabled or disabled for the plan or package.

(IMSMPP)
Specifies the IMS connection for the Message Processing Program (MPP) and IMS Fast Path (IFP) regions. All IMS MPP connections identified by the value of the IMSID on the CTL parameter EXEC. are either enabled or disabled for the plan or package.

(IMSMPP) IMSMPP(*imsid***, ...)**
Specifies the value of IMSID on the CTL parameter EXEC. The IMS MPP connection identified by *imsid* is either enabled or disabled for the plan or package.

(REMOTE)
Indicates that all remote connections are either enabled or disabled for the plan or package.

(REMOTE) REMOTE (*location-name***,...,<** *luname***>,...)** (PACKAGE only)
Specifies that the remote connections identified by the following are either enabled or disabled for the package:

location-name
Specifies the location name of a requesting DBMS that **is** a DB2 UDB for z/OS subsystem.

< *luname***>**
Specifies the logical unit name, as defined to VTAM at the server location, of a requesting DBMS that **is not** a DB2 UDB for z/OS subsystem.

You must bracket a logical unit name with the less than (<) and the greater than (>) characters to differentiate it from a location name.

(RRSAF)
Indicates that the RRS attachment facility connection is either enabled or disabled for the plan or package.

Performance hint: Whenever the plan or package is allocated, DB2 must check the connection type and connection name with the list of enabled or disabled connections. For best performance, keep the list short.

Plans that disable a system: If a plan disables a system, then no packages appended to that plan can run from that system, regardless of the ENABLE/DISABLE options. However, if the same packages are appended to other plans that enable the system, those packages can run from that system under those plans.

Defaults:

Process	Default value
BIND PLAN	ENABLE(*)
BIND PACKAGE	ENABLE(*)
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

Catalog record: Table SYSPKSYSTEM for packages and table SYSPLSYSTEM for plans.

ENCODING	(**ASCII**) (**EBCDIC**) (**UNICODE**) (**ccsid**)	**On: BIND and REBIND PLAN and PACKAGE**

Specifies the application encoding for all host variables in static statements in the plan or package. EBCDIC is the only valid option for a plan or package that was precompiled on DB2 Version 6 or earlier. If you specify ccsid on any plan or package precompiled prior to V7, the value of the ccsid must match the EBCDIC ccsid specified on the installation panel DSNTIPF (the SYSTEM EBCDIC CCSID). You can specify ASCII, UNICODE, or ccsid, where ccsid is a value other than the SYSTEM EBCDIC ccsid for any plan or package precompiled on V7 or later. You might select this option when a data source, such as a terminal emulator, uses a ccsid that is not the same as the SYSTEM EBCDIC ccsid. For example, a user has a terminal emulator with a ccsid of 1047, but the SYSTEM EBCDIC CCSID is 37. In this case, the plan or package being used by that user should be bound with ENCODING (1047).

ENCODING also affects the content of the data that is returned by the SQL statement DESCRIBE. DB2 will return column names, label names, or both (if requested) in the specified application encoding scheme. See *DB2 SQL Reference* for more information about DESCRIBE.

For Unicode information, see the appendix about Unicode support in *DB2 Installation Guide*.

Defaults: The default package application encoding scheme is not inherited from the plan application encoding option. The default for a package that is bound on a remote DB2 UDB for z/OS system is the remote server's default application encoding scheme. Similarly, when a plan or package is run on a remote DB2 UDB for z/OS server, the specified ENCODING option is ignored. Instead, the remote server's encoding scheme is used.

The following statements set the value of the host variable and do not require the package or DBRM to be bound into the plan:

```
SET CURRENT PACKAGE SET = :HV ,
SET :HV = CURRENT PACKAGE SET ,
SET :HV = CURRENT PACKAGE PATH ,
SET CURRENT PACKAGE PATH = :HV
```

The host variable uses the system default application encoding scheme, even when the application is bound with the ENCODING (EBCDIC/UNICODE) bind option.

Process	Default value
BIND PLAN	The system default application encoding scheme that was specified at installation time.
BIND PACKAGE	The system default application encoding scheme that was specified at installation time.
REBIND PLAN	The value that was specified the last time that the plan was bound.
REBIND PACKAGE	The value that was specified the last time that the package was bound.

Product-sensitive Programming Interface

EXPLAIN	(**NO**) (**YES**)	**On BIND and REBIND PLAN and PACKAGE, REBIND TRIGGER PACKAGE**

Obtains information about how SQL statements in the package, or in the member list of the plan, are to execute. It inserts that information into the table *owner*.PLAN_TABLE. *Owner* can be the authorization ID of the owner of the plan or package. Alternatively, the authorization ID of the owner of the plan or package can have an alias as *owner*.PLAN_TABLE that points to the base table, PLAN_TABLE. *Owner* must also have the appropriate SELECT and INSERT privileges on that table. This option does not obtain information for statements that access remote objects.

PLAN_TABLE must have a base table and can have multiple aliases with the same table name, PLAN_TABLE, but using different authids; it cannot be a view or a synonym. It should exist before the bind process begins.

The EXPLAIN option also populates two optional tables, if they exist: DSN_STATEMNT_TABLE and DSN_FUNCTION_TABLE.

DSN_STATEMNT_TABLE contains DB2's estimate of the processing cost for an SQL statement. See Part 6 of *DB2 Application Programming and SQL Guide* for more information.

DSN_FUNCTION_TABLE contains information about function resolution. See Part 3 of *DB2 Application Programming and SQL Guide* for more information.

You can get EXPLAIN output for a statement that is embedded in a program that is bound with EXPLAIN(NO) by embedding the SQL statement EXPLAIN in the program. Otherwise, the value of the EXPLAIN option applies to all explainable SQL statements in the program, and to the fullselect portion of any DECLARE CURSOR statements.

In all inserts to *owner*.PLAN_TABLE, the value of QUERYNO is the statement number that the precompiler assigned and placed in the DBRM.

For a description of the tables populated by the EXPLAIN option, see information about the EXPLAIN statement in Chapter 5 of *DB2 SQL Reference*.

For automatic rebind: EXPLAIN(YES) is in effect if you bind the plan or package with EXPLAIN(YES) and if the value of field EXPLAIN PROCESSING on installation panel DSNTIPO is YES. If EXPLAIN(YES) and VALIDATE(BIND) are in effect and PLAN_TABLE is not correct, the automatic rebind fails.

(NO) Provides no EXPLAIN information.

(YES) Inserts information in the tables populated by EXPLAIN. If *owner*.PLAN_TABLE does not exist at bind time, the value of the option VALIDATE determines the success of the bind operation.

- If the value is BIND, the bind fails.
- If the value is RUN, DB2 checks to see if the table exists again at run time. If it still does not exist, the plan or package cannot run. If it does exist, DB2 inserts information in PLAN_TABLE before the plan or package runs.

If neither or both of the optional tables DSN_FUNCTION_TABLE or DSN_STATEMNT_TABLE exist, or if they are defined incorrectly, the bind does not fail.

Invalidation resulting from an unsuccessful rebind: An unsuccessful rebind generating a return code of greater than 4 invalidates the rebind object and rolls back all changes to the object, leaving it as it was before the rebind attempt. However, if the rebind fails because of either the REBIND option EXPLAIN or the SQL statement EXPLAIN (that is, the PLAN_TABLE does not exist or was created incorrectly), DB2 rolls back all changes to the object but does not invalidate the object.

Defaults:

Process	Default value
BIND PLAN	NO
BIND PACKAGE	NO
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

Catalog record: Column EXPLAIN of table SYSPACKAGE and column EXPLAN of SYSPLAN.

End of Product-sensitive Programming Interface

FLAG	(**I**) (**W**) (**E**) (**C**)	**On: BIND and REBIND PLAN and PACKAGE, REBIND TRIGGER PACKAGE**

Determines what messages to display.

(I) All informational, warning, error, and completion messages

(W) Only warning, error, and completion messages

(E) Only error and completion messages

(C) Only completion messages

Rebinding multiple plans or packages: When your REBIND command contains an asterisk (*) and affects many plans or packages, FLAG(E) is recommended to avoid running out of message storage.

Defaults:

Process	Default value
BIND PLAN	I
BIND PACKAGE	I
REBIND PLAN	I
REBIND PACKAGE	I

IMMEDWRITE	(**NO**) (**YES**)	**On: BIND and REBIND PLAN and PACKAGE**

Indicates whether immediate writes are to be done for updates that are made to group buffer pool dependent page sets or partitions. This option is only applicable for data sharing environments. The IMMEDWRITE subsystem parameter has no effect on the IMMEDWRITE bind option at bind time. Table 15 shows the implied hierarchy of this option as it affects run time. The IMMEDWRITE option values are as follows:

(NO) Specifies that normal write activity is done. Updated pages that are group buffer pool dependent are written at or before phase one of commit or at the end of abort for transactions that have rolled back.

(YES) Specifies that updated pages that are group buffer pool dependent are immediately written as soon as the buffer update completes. Updated pages are written immediately even if the buffer is updated during forward progress or during rollback of a transaction. Specifying this option might impact performance.

Table 15. The implied hierarchy of the IMMEDWRITE option

IMMEDWRITE bind option	IMMEDWRI subsystem parameter	Value at run time
NO	NO	NO
NO	PH1	PH1
NO	YES	YES
PH1	NO	PH1
PH1	PH1	PH1
PH1	YES	YES
YES	NO	YES
YES	PH1	YES
YES	YES	YES

Note: The NO and PH1 options are equivalent. The PH1 option is shown for backward compatibility only.

Performance hints: You can use IMMEDWRITE(PH2) and IMMEDWRITE(YES) for situations where a transaction spawns another transaction that can run on another DB2 member and that depends on uncommitted updates that were made by the originating transaction.

Specify IMMEDWRITE(PH2) to cause group buffer pool dependent pages to be written at or before phase 1 of commit.

Specify IMMEDWRITE(YES) to cause the originating transaction to immediately write its updated GBP-dependent buffers (instead of waiting until the end of commit or rollback), which will ensure that the dependent transaction always gets the same results regardless of whether it runs on the same member or a different member as the originating transaction. IMMEDWRITE(YES) should be used with caution because of its potential impact to performance. The impact will be more significant for plans and packages that do many buffer updates to GBP-dependent pages, and not as noticeable for plans or packages that perform few buffer updates to GBP-dependent pages. The following options can be considered as alternatives to using IMMEDWRITE(YES):

- Always run the dependent transaction on the same DB2 member as the originating transaction.
- Run the dependent transaction with ISOLATION(RR).
- Wait until the completion of phase two of commit before spawning the dependent transaction.
- CURRENTDATA(YES) or ISOLATION(RS) can be used to solve the problem only if the originating transaction updates columns that are not in the WHERE clause of the dependent transaction.

Defaults:

Process	Default value
BIND PLAN	NO
BIND PACKAGE	NO
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

The default for a package on a remote DB2 server is IMMEDWRITE(NO).

ISOLATION (**RR**) (**RS**) (**CS**) (**UR**) (**NC**) — **On: BIND and REBIND PLAN and PACKAGE, REBIND TRIGGER PACKAGE**

Determines how far to isolate an application from the effects of other running applications. For more information on isolation levels, see Improving Concurrency in Part 5 (Volume 2) of *DB2 Administration Guide*.

(RR) *Repeatable read*. Ensures that:

- Your application does not read a row that another process has changed until that process releases that row.

- Other processes do not change a row that your application reads until your application commits or terminates.

(RS) *Read stability*. Ensures that:

- Your application does not read a row that another process has changed until that process releases that row.
- Other processes do not change a row that satisfies the application's search condition until your application commits or terminates. It does allow other application processes to insert a row, or to change a row that did not originally satisfy the search condition.

If the server does not support RS, it uses RR.

(CS) *Cursor stability*. Ensures, like repeatable read, that your application does not read a row that another process changes until that process releases that row. Unlike repeatable read, cursor stability does not prevent other applications from changing rows that your application reads before your program commits or terminates.

(UR) *Uncommitted read*. Unlike repeatable read and cursor stability, does not ensure anything. With the exception of LOB data, uncommitted read avoids acquiring locks on data and allows:

- Other processes change any row your application reads during the unit of work.
- Your application read any row that another process has changed, even if the process has not committed the row.

You can use this option only with a read-only operation: SELECT, SELECT INTO, or FETCH using a read-only cursor. If you specify ISOLATION(UR) for any other operation, DB2 uses ISOLATION(CS) for that operation.

(NC) *No commit*. Used on packages that are bound to certain servers other than DB2 UDB for z/OS. DB2 UDB for z/OS does not support NC. If the server does not support this isolation level, it uses UR.

For more information about how the ISOLATION option affects locking and concurrency, including how DB2 resolves conflicts by using the most restrictive value when the values specified in the plan and package differ, see Part 4 of *DB2 Application Programming and SQL Guide*.

Defaults:

Process	Default value
BIND PLAN	RR
BIND PACKAGE	Plan value
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

The default for binding a package to a remote server is **RR**.

For REBIND PACKAGE, you cannot change ISOLATION from a specified value to a default of the plan value by using REBIND PACKAGE. To do that, you must use BIND PACKAGE ACTION(REPLACE).

Catalog record: Column ISOLATION of tables SYSPACKAGE and SYSPLAN.

KEEPDYNAMIC	(NO) (YES)	On: BIND and REBIND PLAN and PACKAGE

Determines whether DB2 keeps dynamic SQL statements after commit points.

(NO) Specifies that DB2 does not keep dynamic SQL statements after commit points.

(YES) Specifies that DB2 keeps dynamic SQL statements after commit points.

If you specify KEEPDYNAMIC(YES), the application does not need to prepare an SQL statement after every commit point. DB2 keeps the dynamic SQL statement until one of the following occurs:

- The application process ends
- A rollback operation occurs.
- The application executes an explicit PREPARE statement with the same statement identifier.

If you specify KEEPDYNAMIC(YES), and the prepared statement cache is active, DB2 keeps a copy of the prepared statement in the cache. If the prepared statement cache is not active, DB2 keeps only the SQL statement string past a commit point. DB2 then implicitly prepares the SQL statement if the application executes an OPEN, EXECUTE, or DESCRIBE operation for that statement.

If you specify KEEPDYNAMIC(YES), DDF server threads that are used to execute KEEPDYNAMIC(YES) packages will remain active. Active DDF server threads are subject to idle thread timeouts, as described in Part 2 of *DB2 Installation Guide* for installation panel DSNTIPR.

If you specify KEEPDYNAMIC(YES), you must not specify REOPT(ALWAYS). KEEPDYNAMIC(YES) and REOPT(ALWAYS) are mutually exclusive. However, you can use KEEPDYNAMIC(YES) with REOPT(ONCE).

Performance hint: KEEPDYNAMIC(YES) results in improved performance if your DRDA client application uses a cursor defined WITH HOLD. DB2 automatically closes a held cursor when there are no more rows to retrieve, which eliminates an extra network message.

Defaults:

Process	Default value
BIND PLAN	NO
BIND PACKAGE	NO
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

The default for a package on a remote DB2 server is KEEPDYNAMIC(NO).

Catalog record: Column KEEPDYNAMIC of table SYSPLAN and SYSPACKAGE.

LIBRARY	(*dbrm-pds-name*) (*dbrm-pds-name*, ...) (BIND PLAN only)	On: BIND PLAN, BIND PACKAGE

Determines what partitioned data sets (libraries) to search for the DBRMs listed in the MEMBER option. The libraries must be cataloged.

The bind process searches for the libraries in the order that you list them. If the libraries do not contain some DBRM listed in the MEMBER option, and if a JCL statement exists for DBRMLIB DD, then the process searches for the member among the libraries that the JCL statement describes.

dbrm-pds-name is the data set name of a library.

For BIND PACKAGE, you can specify only one library to search.

For BIND PLAN, you can specify one or more libraries to search.

Defaults:

Process	Default value
BIND PLAN	None
BIND PACKAGE	None
REBIND PLAN	N/A
REBIND PACKAGE	N/A

The default is to search only the libraries described by the DD statement for DBRMLIB.

MEMBER	(*dbrm-member-name*) (*dbrm-member-name*, ...) (BIND PLAN only)	**On: BIND PLAN, BIND PACKAGE**

Determines what database request modules (DBRMs) to include in the plan or package.

dbrm-member-name
Specifies the name of a library member that contains a DBRM. You can name the partitioned data set, of which a DBRM is a member, either in the LIBRARY option or in the JCL statement for DBRMLIB DD.

For BIND PACKAGE only, the name becomes the package name. Names beginning with DSN are reserved; you receive a warning message if you use one.

For BIND PACKAGE, you can use only one member. If you do not use MEMBER, you must use COPY. You cannot use both options.

For BIND PLAN, you can list many members. DB2 sorts the member list in alphabetical order. If you do not use MEMBER, you must use PKLIST.

Defaults:

Process	Default value
BIND PLAN	None
BIND PACKAGE	None
REBIND PLAN	N/A
REBIND PACKAGE	N/A

Catalog record: Column NAME of table SYSPACKAGE for BIND PACKAGE, or the table SYSDBRM for BIND PLAN.

OPTHINT ('*hint-id*') **On: BIND and REBIND PLAN and PACKAGE**

Controls whether query optimization hints are used for static SQL.

('*hint-id***')**
A character string of up to 128 characters in length, which is used by the optimizer when searching the PLAN_TABLE for rows to use as input to the optimizer. The delimiters can only be single quotation marks (').

If '*hint-id*' contains all blank characters, DB2 does not use optimization hints for static SQL statements.

DB2 uses optimization hints only when optimization hints are enabled for your system. To enable optimization hints, specify YES in the OPTIMIZATION HINTS field of installation panel DSNTIP4.

For more information about using the OPTHINT option, see Part 5 (Volume 2) of *DB2 Administration Guide*.

Restriction: The PACKAGE does not inherit from the PLAN.

Defaults:

Process	Default value
BIND PLAN	All blanks, use normal optimization
BIND PACKAGE	All blanks, use normal optimization
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

The default for a package on a remote server is all blanks.

Catalog record: Column OPTHINT of tables SYSPLAN and SYSPACKAGE.

OPTIONS **(COMPOSITE)** **(COMMAND)** **On: BIND PACKAGE COPY**

Specifies which bind options to use for the new package.

COMPOSITE
The options for the new package are what you specify on the BIND PACKAGE COPY subcommand. Options that you do not specify are the option values taken from the SYSPACKAGE catalog table row that describes the source package that is to be copied. For a remote copy, OPTIONS(COMPOSITE) is only valid if the remote DB2 subsystem is DB2 UDB for z/OS Version 8.

COMMAND
The options for the new package are what you specify on the BIND PACKAGE COPY subcommand. Options that you do not specify are determined as follows:
- For a local copy, the DB2-defined BIND PACKAGE options defaults are used.

- For a remote copy, the server-defined BIND PACKAGE option defaults are used at the server. You must use OPTIONS(COMMAND) when copying to a down-level server or to a non-z/OS DB2 server. A down-level server is any server that is not DB2 UDB for z/OS Version 8.

Defaults:

Process	Default value
BIND PACKAGE COPY	COMPOSITE

OWNER (*authorization-id*) — **On: BIND and REBIND PLAN and PACKAGE**

Determines the authorization ID of the owner of the object (plan or package). The owner must have the privileges required to execute the SQL statements contained in the object.

If ownership changes, all grants for privileges on the object that the previous owner issued change to name the new owner as the grantor. The new owner has the privileges BIND and EXECUTE on the object and grants them to the previous owner.

You can bind or rebind only the objects for which the authorization ID has bind privileges. If you do not specify an authorization ID, the process rebinds only the objects for which the primary ID has bind privileges.

For remote BIND or REBIND PACKAGE only, the value of OWNER is subject to translation when sent to the remote system.

Defaults:

Process	Default value
BIND PLAN	Primary ID
BIND PACKAGE	Primary ID
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

The default owner is the primary authorization ID of the agent that runs the bind process.

Catalog record: Column OWNER of table SYSPACKAGE, column GRANTOR of table SYSPACKAUTH, and column CREATOR of table SYSPLAN.

PACKAGE (*location-name.collection-id.package-id.(version-id)*) (*) (REBIND PACKAGE only) — **On: BIND and REBIND PACKAGE**

Determines what package or packages to bind or rebind.

You cannot use the BIND PACKAGE subcommand to:
- Bind a package with the same name as an existing trigger package
- Copy a trigger package

The following options identify the location, collection, package name, and version of the package. You can identify a location and collection. For BIND, the DBRM supplies the package ID and version ID if you use the option MEMBER, or those IDs come from the option COPY. For REBIND, you must identify a package name, and you can also supply a version ID.

location-name
The location of the DBMS where the package binds or rebinds and where the description of the package resides. The location name must be defined in catalog table SYSIBM.LOCATIONS. If that table does not exist or if the DBMS is not in it, you receive an error message.

The default is the local DBMS.

collection-id or *
Specifies the collection to contain the package to bind, or that already contains the package to rebind. There is no default.

For REBIND, you can use an asterisk (*) to rebind all local packages with the specified *package-id* in all the collections for which you have bind privileges.

package-id or * **(For REBIND only)**
Specifies the name of the package to rebind, as listed in column NAME of catalog table SYSPACKAGE. There is no default.

You can use an asterisk (*) to rebind all local packages in *collection-id* for which you have bind privileges.

version-id or * **(For REBIND only)**
Specifies the version of the package to rebind, as listed in column VERSION of catalog table SYSPACKAGE.

You can use an asterisk (*) to rebind all local versions of the specified *package-id* in *collection-id* for which you have bind privileges.

Using simply () rebinds the version of the package that is identified by the empty string.

If you omit *version-id*, the default depends on the how you specify *package-id*. If you use * for *package-id*, then *version-id* defaults to *. If you explicitly provide a value for *package-id*, then *version-id* defaults to the empty string version.

DBRMs created in releases of DB2 before Version 2 Release 3 use a *version-id* of the empty string by default.

(*) (For REBIND only)
Rebinds all local DB2 packages for which the applicable authorization ID has the BIND privilege. Specifying (*) is the same as specifying the package name as (*.*.(*)) or (*.*). The applicable authorization ID is:

- The value of OWNER, if you use that option
- The primary authorization ID of the process running the bind, if you do not use the option OWNER

Catalog record: Columns COLLID, NAME, and VERSION of table SYSPACKAGE.

For more information about:

- How to define a location name in SYSIBM.LOCATIONS, see Part 3 of *DB2 Administration Guide*.
- Which packages are bound depending on how you specify collections, packages, and versions on the REBIND PACKAGE command, see Part 4 of *DB2 Application Programming and SQL Guide*.

PATH	(*schema-name*) (**USER**) (*schema-name*,**USER**, ...)	**On: BIND and REBIND PLAN and PACKAGE**

Determines the SQL path that DB2 uses to resolve unqualified user-defined distinct types, functions, and stored procedure names (in CALL statements).

For the **PATH** option, consider the following guidelines when you specify a *schema-name*:

- The specified schema names are *not* folded to uppercase by DB2. This behavior is different than that for schema names in SQL statements, which are folded to uppercase before being stored in the catalog. If you do not specify these nondelimited schema names in upper case, DB2 cannot find a match in the catalog for those schema names.
- You can specify delimited identifiers in both mixed and uppercase characters.

The **PATH** keyword is mutually exclusive with the **PATHDEFAULT** keyword. Do not specify both keywords in the same REBIND command.

(*schema-name***)**
Identifies a schema.

DB2 does not validate that the specified schema actually exists at precompile or at bind time.

You do not need to explicitly specify the SYSIBM, SYSFUN, and SYSPROC schemas; DB2 implicitly assumes that these schemas are at the beginning of the SQL path. DB2 adds these schemas in the order listed. If you do not specify the SYSIBM, SYSFUN, and SYSPROC schemas, they are not included in the 2048-byte length.

(*schema-name***, ...)**
Identifies a list of schemas. The same schema name should not appear more than once in the SQL path.

The number of schemas that you can specify is limited by the length of the resulting SQL path, which cannot exceed 2048 bytes. To calculate the length of the resulting SQL path:

1. Take the length of each schema.
2. Add 2 for delimiters around each *schema-name* in the list.
3. Add 1 for each comma after each schema. Do **not** add 1 for the last schema.

USER Represents a maximum 8-byte *schema-name*. At bind time, DB2 includes this 8-byte length in the total length of the list of schema names specified for the PATH bind option. The maximum length for a list of schema names, including comma separators, delimiters, and the 8-byte USER value, is 2048 bytes. If you exceed this limit, DB2 generates an error message at bind time.

At run time, DB2 substitutes the run-time value of the USER special register, which contains the primary authorization ID of the run-time process, for the *schema-name* in the position of USER in the PATH *schema-name* list.

If you specify USER in a list of schema names, do not use delimiters around the USER keyword.

For more information about schema names, ordinary identifiers, and delimited identifiers, see Chapter 2 of *DB2 SQL Reference*.

Defaults:

Process	Default value
BIND PLAN	"SYSIBM," "SYSFUN," "SYSPROC," *plan qualifier*
BIND PACKAGE	"SYSIBM," "SYSFUN," "SYSPROC," *package qualifier*
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

Although *plan qualifier* is the default value for BIND PLAN, it is not stored in the catalog. Instead, the catalog value is blank. The catalog value is also blank for *package qualifier*.

PATHDEFAULT — **On: REBIND PLAN and PACKAGE**

Resets the **PATH** for a package or plan to "SYSIBM," "SYSFUN," "SYSPROC," or *plan qualifier/package qualifier*.

The **PATHDEFAULT** keyword is mutually exclusive with the PATH keyword. Do not specify both keywords in the same REBIND command.

Defaults:

Process	Default value
BIND PLAN	N/A
BIND PACKAGE	N/A
REBIND PLAN	None
REBIND PACKAGE	None

PKLIST (*location-name.collection-id.package-id*, ...) PKLIST only
NOPKLIST — **On: BIND and REBIND PLAN**

PKLIST determines what packages to include in the package list for the plan. The order in which you list packages with partial identifiers determines the search order at run time and can affect performance.

NOPKLIST is used with REBIND PLAN only. NOPKLIST determines that the plan rebinds without a package list. If a package list already exists, NOPKLIST deletes it.

location-name or *

Names the location of the DBMS where the package resides, or defers that choice until run time. Use either a particular location name or an asterisk (*), or omit this part of the identifier. The default is the local DBMS.

- If you use a particular location name, then that DBMS should be defined in catalog table SYSIBM.LOCATIONS. If that table does not exist or if the DBMS is not in it, you receive warning messages.

- If you use an asterisk, at run time the location comes from the special register CURRENT SERVER. DB2 checks privileges to use the SQL statements in the package at that location.

collection-id or *

Names the collection that contains the package or defers that choice until run time. Use either a particular collection ID or an asterisk (*). No default exists.

If you use an asterisk, then DB2 checks the privileges to use the SQL statements that are embedded in the package at run time. At that time also, DB2 determines the collection ID as follows:

- If the value in the special register CURRENT PACKAGESET is not blank, then that value is the collection ID.
- If the value of CURRENT PACKAGESET is blank, DB2 skips the entry unless it is the last entry in the package list. If it is the last or only entry, an error message is issued.

package-id or *

Names a particular package or specifies, by the asterisk, all packages in the collection. Because you cannot specify a *version-id* for the packages included in the package list, all versions are effectively included.

Defaults:

Process	Default value
BIND PLAN	None
BIND PACKAGE	N/A
REBIND PLAN	Existing value
REBIND PACKAGE	N/A

PKLIST has no default; if you do not use PKLIST, you must use MEMBER.

The default for NOPKLIST is to use the package list specified in the PKLIST option, if any, during the current or previous bind or rebind.

Catalog record: Table SYSPACKLIST.

For more information about:

- How the order of search for packages affects performance, see Part 5 of *DB2 Application Programming and SQL Guide*.
- How to define a location name in SYSIBM.LOCATIONS, see Part 3 of *DB2 Administration Guide*.
- The TSO/E restriction that limits the maximum number of packages specified in the PKLIST, see *z/OS TSO/E Programming Services*.

PLAN (*plan-name*) (*) (REBIND PLAN only) — **On: BIND and REBIND PLAN**

Determines what plan or plans to bind or rebind.

(*plan-name***)**

Specifies the name of the application plan.

For REBIND only, the value of column NAME in the catalog table SYSPLAN; you can use a list of plan names.

The default is to perform all bind functions, including error diagnostics, without producing an application plan and without inserting rows into PLAN_TABLE for the option EXPLAIN.

(*) (For REBIND only)
Rebinds all plans for which the applicable authorization ID has the BIND privilege. The applicable ID is:

- The value of OWNER, if you use that option
- The authorization ID of the process running the bind, if you do not use the option OWNER

Catalog record: Column NAME of table SYSPLAN.

QUALIFIER (*qualifier-name*) **On: BIND and REBIND PLAN and PACKAGE**

Determines the implicit qualifier for unqualified names of tables, views, indexes, and aliases contained in the plan or package.

(*qualifier-name***)**
Specifies the value of the implicit qualifier. This value is not subject to translation when sent to a remote system for BIND or REBIND PACKAGE.

Defaults:

Process	Default value
BIND PLAN	Owner ID
BIND PACKAGE	Owner ID
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

The default is the owner's authorization ID, whether you use the OWNER option or its default.

Catalog record: Column QUALIFIER of tables SYSPACKAGE and SYSPLAN.

RELEASE (**COMMIT**) (**DEALLOCATE**) **On: BIND and REBIND PLAN and PACKAGE, REBIND TRIGGER PACKAGE**

Determines when to release resources that a program uses, either at each commit point or when the program terminates.

(COMMIT)
Releases resources at each commit point.

(DEALLOCATE)
Releases resources only when the program terminates.

RELEASE(DEALLOCATE) has no effect on packages that are run on a DB2 server through a DRDA connection with a client system. The value also has no effect on dynamic SQL statements, which always use RELEASE(COMMIT), with one exception: When you use RELEASE(DEALLOCATE) and KEEPDYNAMIC(YES), and your subsystem is installed with YES for field

CACHE DYNAMIC SQL on installation panel DSNTIP4, the RELEASE(DEALLOCATE) option is honored for dynamic SELECT, INSERT, UPDATE and DELETE statements.

Locks that are acquired for dynamic statements are held until one of the following events occurs:

- The application process ends (deallocation).
- The application issues a PREPARE statement with the same statement identifier. (Locks are released at the next commit point.)
- The statement is removed from the cache because it has not been used. (Locks are released at the next commit point.)
- An object that the statement is dependent on is dropped or altered, or a privilege that the statement needs is revoked. (Locks are released at the next commit point.)

RELEASE(DEALLOCATE) can increase the package or plan size, because additional items become resident in the package or plan.

For more information about how the RELEASE option affects locking and concurrency, see Part 5 (Volume 2) of *DB2 Administration Guide* or Part 5 of *DB2 Application Programming and SQL Guide*.

Defaults:

Process	Default value
BIND PLAN	COMMIT
BIND PACKAGE	Plan value
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

COMMIT is the default for a package that is bound at a remote server.

Catalog record: Column RELEASE of tables SYSPACKAGE and SYSPLAN.

REOPT	(**NONE**) (**ALWAYS**) (**ONCE**)	**On: BIND and REBIND PLAN and PACKAGE**

Specifies whether to have DB2 determine an access path at run time by using the values of host variables, parameter markers, and special registers.

(NONE)
Does not determine an access path at run time. You can use NOREOPT(VARS) as a synonym for REOPT(NONE).

(ALWAYS)
Re-determines the access path at run time each time the statement is run. You can use REOPT(VARS) as a synonym for REOPT(ALWAYS).

(ONCE)
Determines the access path for any dynamic statement only once, at the first run time or at the first time the statement is opened. This access path is used until the prepared statement is invalidated or removed from the dynamic statement cache and needs to be prepared again.

Usage notes:

You cannot use REOPT(ALWAYS) with the following options:

- REOPT(NONE)
- REOPT(ONCE)
- KEEPDYNAMIC(YES)

The following restrictions apply to REOPT(ONCE):

- You can use REOPT(ALWAYS), REOPT(ONCE), REOPT(ONCE) only with DB2 Version 8 new-function mode or later. These REOPT options mark the plan or package with Version 8 dependency. In Version 8 compatibility mode, you must use REOPT(VARS) and NOREOPT(VARS) instead.
- REOPT(ONCE) is ignored if you use it with static SQL statements because DB2 UDB for z/OS caches only dynamic statements.
- If a dynamic statement in a plan or package that is bound with REOPT(ONCE) runs when dynamic statement caching is turned off, the statement runs as if REOPT(ONCE) is not specified.
- You **cannot** use both REOPT(ONCE) and NODEFER(PREPARE).
- You **can** use both REOPT(ONCE) and KEEPDYNAMIC(YES).

Defaults:

Process	Default value
BIND PLAN	NONE
BIND PACKAGE	NONE
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

REOPT(NONE) is the default for a package that is bound on a remote DB2 server.

Catalog record: Column REOPTVAR of table SYSPLAN and SYSPACKAGE.

SQLERROR (**NOPACKAGE**) (**CONTINUE**) **On: BIND PACKAGE only**

Determines whether to create a package if SQL errors occur.

(NOPACKAGE)
Creates no package if an error occurs.

(CONTINUE)
Creates a package, even if errors occur when binding SQL statements. The statements in error cannot execute. Any attempt to execute them at run time causes errors.

Defaults:

Process	Default value
BIND PLAN	N/A
BIND PACKAGE	NOPACKAGE
REBIND PLAN	N/A
REBIND PACKAGE	N/A

Because you cannot use the option SQLERROR for REBIND PACKAGE, the value for the previous package remains in effect when you rebind that package. If you rebind a package that uses SQLERROR(CONTINUE), those SQL statements found in error at bind time do not rebind.

Catalog record: Column SQLERROR of table SYSPACKAGE.

SQLRULES (**DB2**) (**STD**) — **On: BIND and REBIND PLAN**

Determines whether you can execute a type 2 CONNECT statement to an existing SQL connection, according to DB2 rules. Alternatively, the statement causes an error, according to the ANSI/ISO SQL standard of 1992. This option applies to any application process that uses the plan and executes type 2 CONNECT statements. It has no effect on type 1 CONNECT statements or the rules for DB2 private protocol access.

(DB2) No error occurs if CONNECT identifies an existing SQL connection. If X is an existing SQL connection, CONNECT TO X makes X the current connection. If X is already the current connection, CONNECT TO X has no effect on the state of any connections.

(STD) An error occurs if CONNECT identifies an existing SQL connection. Therefore, if X is a dormant SQL connection, you must use the SQL statement SET CONNECTION to make X the current connection.

For local operations, the value of SQLRULES is used for the initial value of the SQL special register CURRENT RULES.

Defaults:

Process	Default value
BIND PLAN	DB2
BIND PACKAGE	N/A
REBIND PLAN	Existing value
REBIND PACKAGE	N/A

Catalog record: Column SQLRULES of table SYSPLAN.

VALIDATE (**RUN**) (**BIND**) — **On: BIND and REBIND PLAN and PACKAGE**

Determines whether to recheck, at run time, errors of the type "OBJECT NOT FOUND" and "NOT AUTHORIZED" found during bind or rebind. The option has no effect if all objects and needed privileges exist.

(RUN) Indicated that if not all objects or privileges exist at bind time, the process issues warning messages, but the bind succeeds. DB2 checks existence and authorization again at run time for SQL statements that failed those checks during bind. The checks use the authorization ID of the plan or package owner.

(BIND)

Indicates that if not all objects or needed privileges exist at bind time, the process issues error messages, and does not bind or rebind the plan or package, *except that*:

For BIND PACKAGE only, if you use the option SQLERROR(CONTINUE), the bind succeeds, but the SQL statements in it that have errors cannot execute.

Defaults:

Process	Default value
BIND PLAN	RUN
BIND PACKAGE	RUN
REBIND PLAN	Existing value
REBIND PACKAGE	Existing value

Catalog record: Column VALIDATE of tables SYSPACKAGE and SYSPLAN.

Chapter 16. -CANCEL THREAD (DB2)

The DB2 command CANCEL THREAD cancels processing for specific local or distributed threads.

Abbreviation: -CAN THD

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage notes" on page 98
- "Examples" on page 101

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or a CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

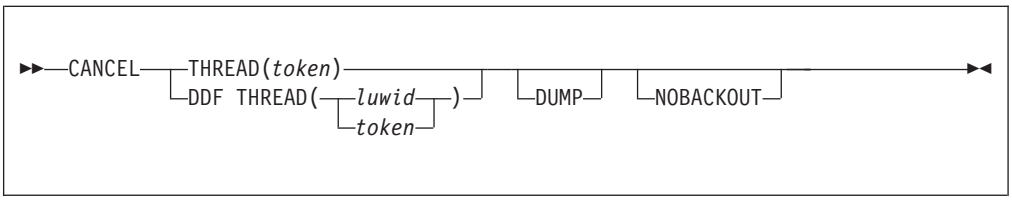
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Option descriptions

THREAD (*token***)**

Identifies a specific thread, either distributed or not, whose processing you want to cancel. DB2 assigns a token to each thread that is unique for that DB2 subsystem, but not necessarily unique across subsystems.

© Copyright IBM Corp. 1983, 2008

The token is a one- to six-digit decimal number. You can determine what the token is by issuing the DB2 command DISPLAY THREAD or by using an IFI READS call for IFCID 0147 or 0148. The token can also appear after the equal sign in DB2 messages that display an LUWID.

DDF THREAD(*luwid*)
Identifies distributed threads for which you want to cancel processing. *luwid* is a logical unit of work identifier (LUWID), consisting of:

- A fully qualified LU network name, which consists of:
  - A one- to eight-character network ID
  - A period
  - A one- to eight-character network LU name
- An LUW instance number, which consists of 12 hexadecimal characters that uniquely identify the unit of work

If you enter three fields separated by periods, DB2 assumes that you are entering an LUWID.

You might have two or more distributed threads with the same LUWID. All distributed threads with the same LUWID are canceled.

The LUWID can be determined from the DB2 DISPLAY THREAD command and other DB2 messages.

DUMP
Provides a dump for diagnostic purposes.

When you cancel a thread that is not currently active in DB2, DB2 performs a hard cancel and no dump is provided. A thread is considered to be not currently active in DB2 when it has left DB2 to perform application work.

NOBACKOUT
Specifies that DB2 is not to attempt to back out the data during transaction rollback processing. Canceling the thread with NOBACKOUT leaves objects in an inconsistent state. Do not issue this command with NOBACKOUT unless you have a plan to resolve the data inconsistency.

Multiple NOBACKOUT requests are allowed. However, if the thread is active and the request is accepted, subsequent requests are ignored. You can choose to issue a subsequent request if a request fails (as indicated by message DSNI032I). Objects that the thread modifies are recovered (backed out). If back out processing fails, the objects are marked REFRESH PENDING (REFP) and either RECOVER PENDING (RECP) or REBUILD PENDING (RBDP or PSRBD) in the database exception table. Resolve the REFP status of the object by running the RECOVER utility to recover the object to a prior point in time or by running LOAD REPLACE on the object.

Usage notes

Canceling distributed threads: Canceling a distributed thread can cause the thread to enter the indoubt state. Message DSNL450I is issued if the CANCEL command causes the DDF thread to be converted from active to indoubt. DB2 releases the resources that the thread holds when the indoubt state is resolved by automatic indoubt resolution with the coordinator, or by resolution with the command RECOVER INDOUBT.

If a thread that is specified in the command is part of a global transaction, the command is executed against all threads in the global transaction. See Chapter 5 of *DB2 Administration Guide*, for an explanation of global transactions.

The CANCEL command schedules a thread to be terminated in DB2. To terminate, the thread must be processing within DB2. If the thread does not terminate, it could be:

- Processing outside of DB2, possibly in the application. If that is the case, the thread does not terminate until the application makes a request to DB2. Use the z/OS CANCEL command to terminate the application immediately.
- Hung up in a network operation. Use VTAM or TCP/IP commands to cause the network operation to return processing to DB2, which will then allow the thread to be terminated. See the following topic for details.

Canceling local threads: The CANCEL command schedules a thread to termintate. Threads that are not in DB2 terminate immediately.

Using VTAM commands to cancel SNA distributed threads: If the CANCEL command does not terminate a distributed thread, it is possible that it is hung up in VTAM. Use the VTAM VARY NET,TERM command to cancel the thread's VTAM sessions. To do this, you need to know the VTAM session IDs (SIDs) that correspond to the thread. Take the following steps:

1. Issue the DB2 command DISPLAY THREAD(*) LUWID(nnnn) DETAIL. (The value of *nnnn* is the token or LUWID provided by CANCEL DDF THREAD.)

   This gives you the VTAM session IDs that must be canceled. Sessions are identified by the column header SESSID as shown in the following DISPLAY THREAD output:

   ```
   -DIS THD(*) LUWID(123) DETAIL
   DSNV401I - DISPLAY THREAD REPORT FOLLOWS:
   DSNV402I - ACTIVE THREADS:
   NAME     ST A   REQ ID           AUTHID    PLAN      ASID TOKEN
   BATCH    TR *     5 BKH2C        SYSADM    BKH2      000D   123
    V444-DB2NET.LUND0.9F6D9F459E92=123 ACCESSING DATA AT
    V446-SAN JOSE:LUND1
    V447--LOCATION         SESSID             A ST    TIME
    V448--SAN JOSE         00D3590EA1E89701     S1    9332108460302
    V448--SAN JOSE         00D3590EA1E89822   V R1    9332108460431
   DISPLAY ACTIVE REPORT COMPLETE
   DSN9022I - DSNVDT '-DIS THD' NORMAL COMPLETION
   ```

   The **V** indicates the thread is processing in VTAM.

2. Record positions 3 through 16 of SESSID for the threads to be canceled. (In the preceding DISPLAY THREAD output, the values are D3590EA1E89701 and D3590EA1E89822.)
3. Issue the VTAM command DISPLAY NET to display the VTAM session IDs. The ones you want to cancel match the SESSIDs in positions 3 through 16 and the corresponding session IDs are in bold. The following is an output example of this command:

```
D NET,ID=LUND0,SCOPE=ACT

IST097I DISPLAY ACCEPTED
IST075I NAME = LUND0, TYPE = APPL
   IST486I STATUS= ACTIV, DESIRED STATE= ACTIV
   IST171I ACTIVE SESSIONS = 0000000005, SESSION REQUESTS = 0000000000
   IST206I SESSIONS:
   IST634I NAME     STATUS          SID          SEND   RECV  VR  TP  NETID
   IST635I LUND1    ACTIV-S     D24B171032B76E65  0051   0043   0   0  NET2
   IST635I LUND1    ACTIV-S     D24B171032B32545  0051   0043   0   0  NET2
   IST635I LUND1    ACTIV-R     D2D3590EA1E89701  0022   0031   0   0  NET2
   IST635I LUND1    ACTIV-R     D2D3590EA1E89802  0022   0031   0   0  NET2
   IST635I LUND1    ACTIV-R     D2D3590EA1E89822  0022   0031   0   0  NET2
   IST314I END
```

4. Issue the VTAM command VARY NET,TERM for each of the VTAM SIDs associated with the DB2 thread. In this case, you might need to cancel only the session ID that DISPLAY THREAD shows to be processing in VTAM (D2D3590EA1E89822).

For more information about VTAM commands, see *VTAM for MVS/ESA Operation*.

Using TCP/IP Commands to Cancel TCP/IP Distributed Threads: If the CANCEL command does not terminate a distributed thread, the thread might be hung up in TCP/IP. Use the TCP/IP DROP command to cancel the thread's connection ID. To do this, you need to first determine the TCP/IP connection ID that corresponds to the thread. Depending on whether the thread is a DB2 requester or server thread, take the following steps:

- Terminating TCP/IP connection for a requester thread:
  1. Issue the DB2 command DISPLAY THREAD(*) LUWID(nnnn) DETAIL. (The value of *nnnn* is the token or LUWID provided by CANCEL THREAD.)

     Find the IP address and local port for the connection to the partner, as shown in the following DISPLAY THREAD output:

     ```
     #display thread(*) detail

     DSNV401I  # DISPLAY THREAD REPORT FOLLOWS -
     DSNV402I  # ACTIVE THREADS -
     NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
     TEST0001 TR       4 CTHDCORID001 SYSADM   DONSQL1  0027    19
      V444-USIBMSY.SYEC715B.B4FA989AF056=19 ACCESSING DATA AT
      V446-STL714A:9.112.114.102:446
      V447--LOCATION         SESSID           A ST TIME
      V448--STL714A          1028:446         V R2 0032608521413
     DISPLAY ACTIVE REPORT COMPLETE
     DSN9022I  # DSNVDT '-DIS THD' NORMAL COMPLETION
     ```

     In this case, the partner's IP address and port is 9.112.114.102 446, and the local port is 1028. V indicates that the thread is processing in TCP/IP.
  2. Determine the associated TCP/IP connection ID:

     ```
     d tcpip,,netstat,conn,ipaddr=9.112.114.102

     EZZ2500I NETSTAT CS V2R10 TCPIP
     USER ID  CONN     LOCAL SOCKET           FOREIGN SOCKET         STATE
     V71BDIST 0000049D 9.112.114.103..1028    9.112.114.102..446     ESTBLSH
     1 OF 1 RECORDS DISPLAYED
     ```
  3. Terminate the connection:

     ```
     v tcpip,,drop,conn=0000049d

     EZZ0060I PROCESSING COMMAND: VARY TCPIP,,DROP,
      CONN=0000049D
     EZZ0053I COMMAND VARY DROP COMPLETED SUCCESSFULLY
     ```
- Terminating TCP/IP connection for a server thread::
  1. Issue the DB2 command DISPLAY THREAD(*) LUWID(nnnn) DETAIL. (The value of *nnnn* is the token or LUWID provided by CANCEL THREAD.)

     Find the IP address and local port for the connection to the partner, as shown in the following DISPLAY THREAD output:

     ```
     !display thread(*) detail

     DSNV401I ! DISPLAY THREAD REPORT FOLLOWS -
     DSNV402I ! ACTIVE THREADS -
     NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
     TEST0001 RA *     2 CTHDCORID001 SYSADM   DONSQL1  002D    11
      V445-USIBMSY.SYEC715B.B4FA9BB94FA7=11 ACCESSING DATA FOR
           9.112.114.103
     ```

```
 V447--LOCATION         SESSID           A ST TIME
 V448--9.112.114.103    446:1029         W R2 0032609061159
DISPLAY ACTIVE REPORT COMPLETE
DSN9022I ! DSNVDT '-DIS THD' NORMAL COMPLETION
```

In this case, the partner's IP address is 9.112.114.103 and the local port is 1029.

2. Determine the associated TCP/IP connection ID:

```
d tcpip,,netstat,conn,ipaddr=9.112.114.103

EZZ2500I NETSTAT CS V2R8 TCPIP
USER ID  CONN     LOCAL SOCKET           FOREIGN SOCKET         STATE
V61ADIST 0000048E 9.112.114.102..446     9.112.114.103..1029    ESTABLS
1 OF 1 RECORDS DISPLAYED
```

Find the entry where the foreign socket shows the partner's IP address and port (9.112.114.103 1029) and note the CONN.

3. Terminate the connection:

```
v tcpip,,drop,conn=0000048e

EZZ0060I PROCESSING COMMAND: VARY TCPIP,,DROP,
 CONN=0000048E
EZZ0053I COMMAND VARY DROP COMPLETED SUCCESSFULLY
```

Examples

Example 1: To cancel a non-distributed thread whose token you found through the DISPLAY THREAD command and to produce a diagnostic dump, issue:

```
-CANCEL THREAD (123) DUMP
```

Example 2: To cancel a distributed thread whose LUWID you found through the DISPLAY THREAD command, issue:

```
-CANCEL DDF THREAD (LUDALLAS.DB2SQL1.3042512B6425)
```

Assume that the output from -DISPLAY THREAD shows that the thread-ID and token associated with this LUWID is 45162. You can also cancel this thread by issuing either of the following commands:

```
-CANCEL DDF THREAD (45162)
-CANCEL THREAD (45162)
```

As in the first example, specifying DUMP with any of the commands shown in this example causes a diagnostic dump to be produced.

Chapter 17. /CHANGE (IMS)

The IMS command /CHANGE resets an indoubt unit of recovery as identified by the OASN keyword of the /DISPLAY command. That command deletes the item from the standpoint of IMS, but it does not communicate to DB2.

Abbreviation: /CHA

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage note" on page 104
- "Examples" on page 104

Environment

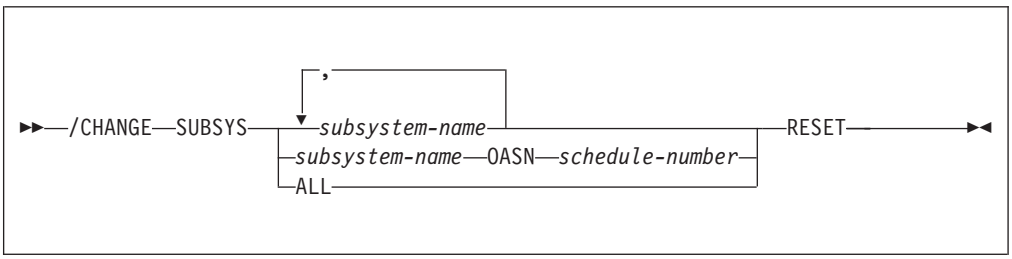
This command can be issued only from an IMS terminal.

Data sharing scope: Member

Authorization

This command requires an appropriate level of IMS authority, as described in *IMS Administration Guide: System*.

Syntax



Option descriptions

SUBSYS
Deletes IMS recovery elements from one or more subsystems. You must code one of the following subparameters:

subsystem-name
Specifies one or more subsystems, seperated by commas, from which recovery elements will be deleted.

© Copyright IBM Corp. 1983, 2008

subsystem-name **OASN** *schedule-number*
Deletes one or more origin application schedule numbers, seperated by commas, from one subsystem, specified by *subsystem-name*.

schedule-number can be a list of up to 32768 origin application schedule numbers. The numbers are displayed using the OASN parameter of the /DISPLAY command.

ALL
Deletes IMS recovery elements from all subsystems.

RESET
Deletes the indoubt recovery unit. The recovery unit represents an incomplete unit of work assigned to an external subsystem as the result of an application request.

Usage note

The preceding description of the /CHANGE command is a partial description only. For a complete description, see *IMS Command Reference*.

Examples

Example 1: Reset all indoubt recovery units for subsystem DB2.

```
/CHA SUBSYS DB2 RESET
```

Example 2: Reset all indoubt recovery units for all subsystems.

```
/CHA SUBSYS ALL RESET
```

Example 3: Reset indoubt recovery units identified by OASN numbers 99, 685, and 2920 for subsystem DB2.

```
/CHA SUBSYS DB2 OASN 99 685 2920 RESET
```

Chapter 18. DCLGEN (DECLARATIONS GENERATOR) (DSN)

The declarations generator (DCLGEN) produces an SQL DECLARE TABLE statement and a COBOL, PL/I, or C data declaration for a table or a view named in the catalog.

For further information regarding the DCLGEN command and uses for its output, see Part 2 of *DB2 Application Programming and SQL Guide*.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 106
- "Option descriptions" on page 106
- "Usage notes" on page 110
- "Examples" on page 112

Environment

The declarations generator is executed by the DSN subcommand DCLGEN. That subcommand can be issued from a DSN session, running in either foreground or background mode, or it can be issued through DB2I.

Data sharing scope: Group

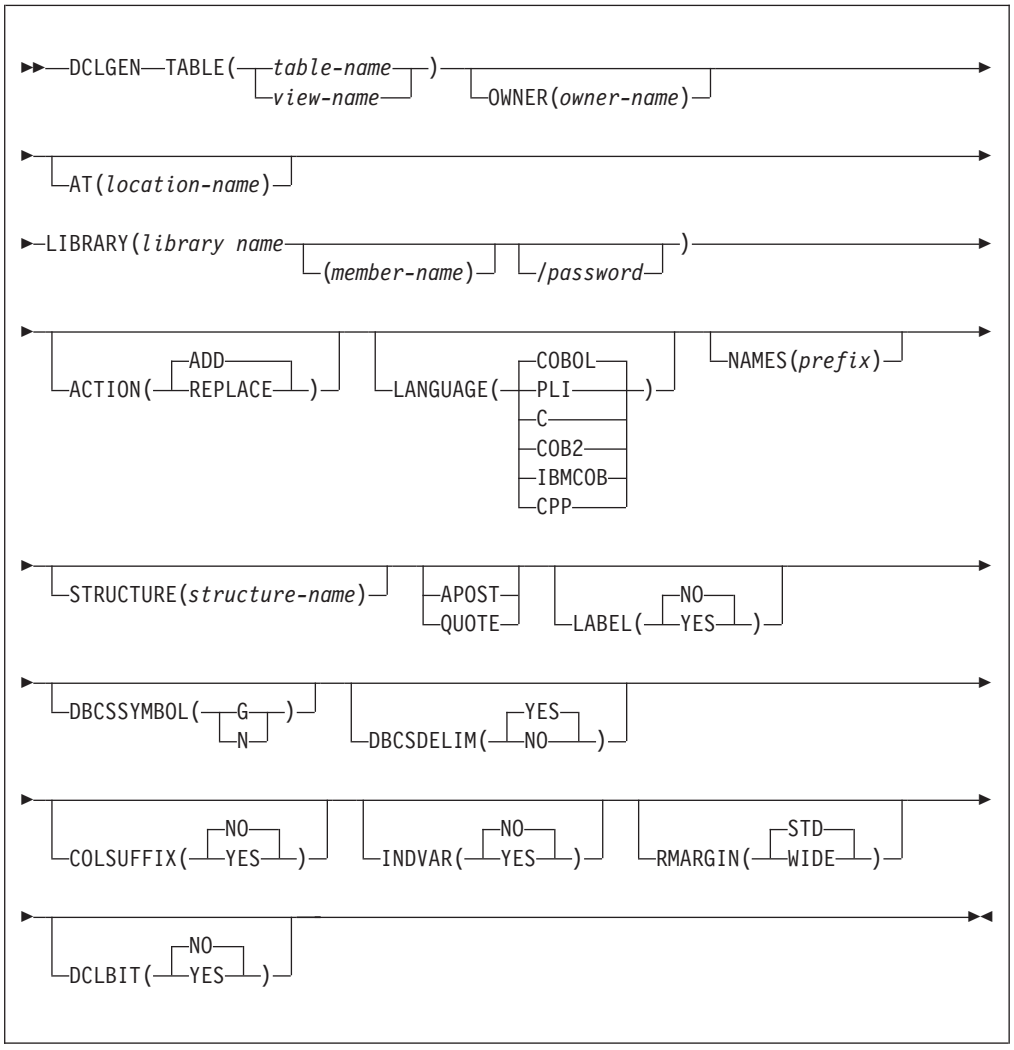
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- Ownership of the table or view
- SELECT privilege on the table or view
- DBADM authority on the database containing the table
- SYSADM authority
- SYSCTRL authority (catalog tables only)

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

TABLE

Specifies the table or view for which a declaration is generated. *table-name* or *view-name* is the qualified or unqualified name of the table or view.

The name must follow these rules:

- If the name is a single-byte or mixed string and contains special characters other than underscores (_), it must be enclosed between apostrophes ('). If the language is COBOL, single-byte underscores in the name are translated into hyphens (-) by DCLGEN. Double-byte character set (DBCS) names need not be enclosed in apostrophes.
- If the name contains single-byte apostrophes, each one must be doubled (''). (Some host languages do not permit apostrophes in variable names.)

A table or view name that contains a period and is not enclosed by apostrophes is a qualified table name. The characters to the left of the period constitute the table owner, and those to the right of the period constitute the table name. Any table name enclosed in apostrophes is an unqualified table

name. To understand how DCLGEN determines the table name qualifier, see the description of the OWNER option, which follows.

OWNER(*owner-name***)**
Specifies a qualifier for the table name. *owner-name* is the qualifier for the table name.

If you specify a qualified table name for the TABLE(*table-name*) option, and you also specify OWNER(*owner-name*), the qualifier portion of *table-name* supersedes *owner-name* as the table name qualifier. If you specify an unqualified table name for the TABLE(*table-name*) option, and you do not specify OWNER(*owner-name*), the SQL authorization ID is the table name qualifier.

DCLGEN supports the use of underscore (_) as a valid character in the *owner-name* keyword parameter.

Table 16 illustrates the decision process for determining the DCLGEN table name qualifier.

Table 16. Decision process for determining the DECLGEN table name qualifier

Table name	**OWNER(***owner-name***) specified**	**OWNER(***owner-name***) not specified**
TABLE(*table-name*) qualified	*table-name* qualifier	*table-name* qualifier
TABLE(*table-name*) unqualified	*owner-name*	SQL authorization ID

AT(*location-name***)**
Identifies the location of the table or view name specified in TABLE (*table-name*). *location-name*, which can consist of 1 to 16 characters, uniquely identifies an instance of a table or view in a network.

If you specify AT, *location-name* is used as the prefix for the table name, and *table-name* or *table-view* must be a qualified name.

DCLGEN supports the use of underscore (_) as a valid character in the *location-name* keyword parameter.

LIBRARY(*library-name***(***member-name***)/***password***)**
Specifies the data set into which the declarations go. This data set must already exist and be accessible to the declarations generator. It can be either sequential or partitioned. *password* is optional.

If the library name is not enclosed within apostrophes, DCLGEN constructs the following full data set name:

```
user-prefix.library-name.language.(member-name)
```

where:

user-prefix	The user prefix of the primary authorization ID of the transaction.
language	The value of the LANGUAGE option: COBOL, COB2, PLI, or C;
(*member-name*)	Optional; if not used, the output goes to a sequential data set.

ACTION
Indicates whether to add or replace the data set.

(ADD)
Adds the data set as a new member, if it does not already exist.

The **default** is **ACTION**(**ADD**).

(REPLACE)
Replaces an existing member or data set with the new one. If the output is to a partitioned data set, and no member exists with the given name, one is added.

LANGUAGE
Specifies the language of the generated declaration.

Possible languages are:

- (**COBOL**), for OS/VS COBOL
- (**COB2**), for other COBOL languages
- (**PLI**), for PL/I
- (**C**), for C/370™
- (**IBMCOB**), for IBM COBOL
- (**CPP**), for C++

NAMES(*prefix***)**
Allows field names to be formed in the declaration.

Avoid possible name conflicts between DCLGEN output and the source program. If a conflict occurs, use NAMES or STRUCTURE, or manually edit the generated declaration or source program.

prefix can contain double-byte characters.

The field names consist of *prefix* concatenated with a number from one to three digits in length. *prefix* can have up to 28 characters. If *prefix* is a single-byte or mixed string and the first character is not alphabetic, it must be enclosed in apostrophes. For example, if *prefix* is ABCDE, the field names will be ABCDE1, ABCDE2, and so on, up to a maximum of ABCDE999. Special characters can be used, but use caution to avoid possible name conflicts.

For COBOL and PL/I, if the prefix is a DBCS string, the field name will be the DBCS prefix concatenated with the DBCS representation of the number. For example, if *prefix* is <D1D2D3> (where “<” and “>” represent shift-out and shift-in characters, respectively, and D1D2D3 represent double-byte characters), generated field names will be <D1D2D3.1>, <D1D2D3.2>, and so on. The period (.) represents X'42'.

The column names in the table are taken as default names for the fields in the output.

STRUCTURE(*structure-name***)**
Specifies the generated data structure.

structure-name can have up to 31 characters. If *structure-name* is a single-byte or mixed string and the first character is not alphabetic, it must be enclosed in apostrophes. You can use special characters, but use caution to avoid possible name conflicts.

structure-name can contain double-byte characters.

For SQL output, the name is the same as the table or view name. If the host language is C, the default structure name is the prefix DCL concatenated with the table name. If the host language is COBOL or PL/I and the table name is a single-byte or mixed string, the default structure name is also the prefix DCL

concatenated with the table name. If the host language is COBOL or PL/I and the table name is a DBCS string, the default structure name is the prefix <.D.C.L> concatenated with the table or view name. “<” and “>” represent shift-out and shift-in characters, respectively. You must guard against possible conflicts with names in the source program. DCLGEN allows the specified structure name to be the same as the table or view name, but will issue a warning message.

APOST or QUOTE
Specifies the string delimiter character used in the host language. This option is effective only for COBOL programs.

APOST specifies the apostrophe (') as the host language string delimiter; the SQL delimiter is the quotation mark (").

QUOTE specifies the quotation mark (") as the host language delimiter; the SQL delimiter is the apostrophe (').

If neither APOST nor QUOTE is specified, the **default** is either APOST or QUOTE for COBOL, depending on what was specified on DB2 installation panel DSNTIPF.

The string delimiter delimits strings in host language statements. The SQL escape character delimits table and column names in the SQL DECLARE TABLE statement produced by DCLGEN. It is possible, by a choice made during DB2 installation, to make both delimiters the quotation mark or both the apostrophe.

LABEL
Indicates whether to include column labels in the output as comments. (Column labels can be assigned by the LABEL ON statement.)

(NO) Omits the column labels.

(YES) Includes the column labels.

DBCSSYMBOL
Specifies the symbol used to denote a graphic data type in a COBOL PICTURE clause.

(G) Graphic data is denoted using G.

(N) Graphic data is denoted using N.

DBCSDELIM
Specifies whether the DBCS table and column names in the generated DECLARE table statement will be delimited.

(YES) DBCS table and column names will be delimited in the DCLGEN table declaration.

(NO) DBCS table and column names will not be delimited in the DCLGEN table declaration.

COLSUFFIX
Determines whether to form field names by attaching the column name to the prefix given by the NAMES option.

(NO)
The column name is not used as a suffix, and field names are controlled by the option NAMES.

(YES)
If NAMES is specified, DCLGEN forms field names by adding column

names as a suffix to the value of NAMES. For example, if the prefix given by NAMES is “NEW” and the column name is EMPNO, the field name is “NEWEMPNO”.

If NAMES is **not** specified, DCLGEN issues a warning message and uses the column names as the field names.

INDVAR
Determines whether to create an indicator variable array for the host variable structure.

(NO)
DCLGEN does not create an indicator variable array.

(YES)
DCLGEN creates an indicator array for the host variable structure. The array name is the table name with a prefix of “I” (or DBCS letter “<I>” if the table name is double-byte).

RMARGIN
Specifies the breakpoint for statement tokens that must be wrapped onto one or more subsequent records of DCLGEN output.

(STD)
Statement tokens will be wrapped after column 72.

(WIDE)
Statement tokens will be wrapped after column 80.

DCLBIT
Specifies whether to generate a DECLARE VARIABLE statement of SQL variables for columns that are defined as FOR BIT DATA. This statement is required in IBM COBOL applications that have host variables for FOR BIT DATA columns and are prepared by using the SQLCCSID option of the integrated DB2 coprocessor. The statement is also valid, but not currently required, for C/C++ and PL/I, and for COBOL that is not prepared by using the SQLCCSID option of the integrated DB2 coprocessor.

(NO)
Does not generate a DECLARE VARIABLE statement of SQL variables for columns that are defined as FOR BIT DATA.

(YES)
Generates a DECLARE VARIABLE statement of SQL variables for columns that are defined as FOR BIT DATA. If the table or view does not have any FOR BIT DATA columns, the statement is not generated.

Usage notes

Parsing of the DCLGEN command conforms to standard TSO parsing conventions. For information about TSO command parsing, see *z/OS TSO/E Programming Services*.

The DECLARE statement: The DECLARE statement generated by DCLGEN will define all columns created with a data type of VARCHAR or LONG VARCHAR as VARCHAR. Columns created with a data type of VARGRAPHIC or LONG VARGRAPHIC will be defined as VARGRAPHIC.

Comments: The output for all host languages includes comments. The leading comment block echoes the DCLGEN subcommand that requested the declarations. The trailing comment block indicates the number of variables declared.

Using the output: To include the DCLGEN output in an application program, use the SQL INCLUDE statement. The same member name specified in the DCLGEN LIBRARY parameter is specified on the INCLUDE statement.

Prompts: Online TSO will prompt for missing or incorrectly specified options.

Editing the output: It is expected that the output of DCLGEN will not meet every need. You can freely edit the output before including it in a program. For example, you might want to change a variable name, or include SQL escape characters.

You can edit the output to add WITH DEFAULT to NOT NULL for columns that do not allow null values. If you edit the output, you must provide a default value.

If your column names contain embedded blanks, they will also be reflected in the host variable declarations, and you will have to remove, or translate, any blank characters to some other value.

C: DCLGEN support of the C language is unique in the following ways:

- DCLGEN does not fold the STRUCTURE, NAMES, or TABLE values to uppercase.
- For any DB2 column that has the data type CHAR(*n*), where $n > 1$, DCLGEN generates the corresponding host variable as CHAR($n + 1$) to avoid the DB2 warning. For $n = 1$, the corresponding host variable is CHAR.

COBOL and binary integers: DB2 uses the full size of binary integers. It can place larger values than allowed in the specified number of digits in the COBOL declaration, which can result in truncated values.

For small integers that can exceed 9999, use S9(5). For large integers that can exceed 999999999, use S9(10) COMP-3 to obtain the decimal data type. If COBOL is used for integers that exceed the COBOL PICTURE, specify the column as decimal to ensure that the data types match and perform well.

COBOL and the underscore character: Because COBOL does not allow the use of the underscore character, DCLGEN translates any underscore characters in the table's column names into hyphens (-) for use in the generated structure.

COBOL and DBCS: OS/VS COBOL does not support DBCS, but later versions of COBOL (VS COBOL II and COBOL/370) do. Although DB2 accepts values outside of the range from X'41' to X'FE', in COBOL data definition statements, both bytes of each double-byte character in data names must be within this range. Data names must also contain at least one DBCS character that does not have X'42' as its first byte.

Data declarations for arrays of indicator variables: If DCLGEN creates an array of indicator variables, data declarations have the following form:

Language	Data declaration
C	short int I*table-name*[*n*];
Cobol	01 I*table-name* PIC S9(4) USAGE COMP OCCURS *n* TIMES
PL/I	DCL I*table-name*(*n*) BIN FIXED (15);

Where *n* is the number of columns in the table.

Examples

Example 1: This example shows the use of the DCLGEN. Issue the following statement from a DSN session or through DB2I:

```
DCLGEN TABLE(VEMPL) -
       LIBRARY('prefix.SRCLIB.DATA(DSN8MPEM)') -
       LANGUAGE(PLI) -
       APOST
```

This statement produces the following statements in *prefix*.SRCLIB.DATA(DSN8MPEM):

```
/************************************************************************/
/* DCLGEN TABLE(VEMPL) -                                                */
/*        LIBRARY('prefix.SRCLIB.DATA(DSN8MPEM)') -                      */
/*        LANGUAGE(PLI) -                                               */
/*        APOST                                                         */
/* ... IS THE DCLGEN COMMAND THAT MADE THE FOLLOWING STATEMENTS         */
/************************************************************************/
EXEC SQL DECLARE VEMPL TABLE
         ( EMPNO                         CHAR(6) NOT NULL,
           FIRSTNME                      VARCHAR(12) NOT NULL,
           MIDINIT                       CHAR(1) NOT NULL,
           LASTNAME                      VARCHAR(15) NOT NULL,
           WORKDEPT                      CHAR(3) NOT NULL
         ) ;
/************************************************************************/
/* PLI DECLARATION FOR TABLE VEMPL                                      */
/************************************************************************/
DCL 1 DCLVEMPL,
     5 EMPNO    CHAR(6),
     5 FIRSTNME CHAR(12) VAR,
     5 MIDINIT  CHAR(1),
     5 LASTNAME CHAR(15) VAR,
     5 WORKDEPT CHAR(3);
/************************************************************************/
/* THE NUMBER OF COLUMNS DESCRIBED BY THIS DECLARATION IS 5             */
/************************************************************************/
```

Example 2: This example shows the use of NAMES and STRUCTURE. Issue the following statement from a DSN session or through DB2I:

```
DCLGEN TABLE(VEMPL) -
       LIBRARY('prefix.SRCLIB.DATA(DSN8MPEM)') -
       LANGUAGE(PLI) -
       NAMES(FIELD) -
       STRUCTURE(EMPRECORD) -
       APOST
```

This statement produces the following statements in *prefix*.SRCLIB.DATA(DSN8MPEM):

```
/*********************************************************************/
/* DCLGEN TABLE(VEMPL) -                                             */
/*        LIBRARY('prefix.SRCLIB.DATA(DSN8MPEM)') -                   */
/*        LANGUAGE(PLI) -                                            */
/*        NAMES(FIELD) -                                             */
/*        STRUCTURE(EMPRECORD) -                                     */
/*        APOST                                                      */
/* ... IS THE DCLGEN COMMAND THAT MADE THE FOLLOWING STATEMENTS      */
/*********************************************************************/
EXEC SQL DECLARE VEMPL TABLE
         ( EMPNO                         CHAR(6) NOT NULL,
           FIRSTNME                      VARCHAR(12) NOT NULL,
           MIDINIT                       CHAR(1) NOT NULL,
           LASTNAME                      VARCHAR(15) NOT NULL,
           WORKDEPT                      CHAR(3) NOT NULL
         ) ;
/*********************************************************************/
/* PLI DECLARATION FOR TABLE VEMPL                                   */
/*********************************************************************/
DCL 1 EMPRECORD,
     5 FIELD1   CHAR(6),
     5 FIELD2   CHAR(12) VAR,
     5 FIELD3   CHAR(1),
     5 FIELD4   CHAR(15) VAR,
     5 FIELD5   CHAR(3);
/*********************************************************************/
/* THE NUMBER OF COLUMNS DESCRIBED BY THIS DECLARATION IS 5          */
/*********************************************************************/
```

Example 3.The following example creates a simple table with columns that are defined as FOR BIT DATA:

```
CREATE TABLE MYTABLE
 ( COL1                 CHAR(10)    NOT NULL
  ,COL2                 CHAR(10)    FOR BIT DATA
  ,COL3                 VARCHAR(12) NOT NULL
  ,COL4                 VARCHAR(12) FOR BIT DATA NOT NULL
```

This statement produces the following results:

```
#                    ************************************************************
#                    * DCLGEN TABLE(MYTABLE)                                    *
#                    *        LIBRARY('prefix.SRCLIB.DATA(MYTABLE)')             *
#                    *                                                         *
#                    *        LANGUAGE(COBOL)                                  *
#                    *        APOST                                            *
#                    *        DCLBIT(YES)                                      *
#                    * ... IS THE DCLGEN COMMAND THAT MADE THE FOLLOWING STATEMENTS   *
#                    ************************************************************
#                         EXEC SQL DECLARE MYTABLE TABLE
#
#                         ( COL1                          CHAR(10) NOT NULL,
#                           COL2                          CHAR(10),
#                           COL3                          VARCHAR(12) NOT NULL,
#                           COL4                          VARCHAR(12) NOT NULL
#                         ) END-EXEC.
#                    ************************************************************
#                    * DECLARED VARIABLES FOR 'FOR BIT DATA' COLUMNS            *
#                    ************************************************************
#                         EXEC SQL DECLARE
#                          :COL2
#                         ,:COL4
#                         VARIABLE FOR BIT DATA END-EXEC.
#                    ************************************************************
#                    * COBOL DECLARATION FOR TABLE MYTABLE                      *
#                    ************************************************************
#                     01  DCLMYTABLE.
#                         10 COL1                 PIC X(10).
#                         10 COL2                 PIC X(10).
#                         10 COL3.
#                            49 COL3-LEN          PIC S9(4) USAGE COMP.
#                            49 COL3-TEXT         PIC X(12).
#                         10 COL4.
#                            49 COL4-LEN          PIC S9(4) USAGE COMP.
#                            49 COL4-TEXT         PIC X(12).
#                    ************************************************************
#                    * THE NUMBER OF COLUMNS DESCRIBED BY THIS DECLARATION IS 4  *
#                    ************************************************************
```

Chapter 19. /DISPLAY (IMS)

The IMS command /DISPLAY displays the status of the connection between IMS and an external subsystem (as well as all application programs communicating with the external subsystem), or the outstanding recovery units that are associated with the subsystem.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Output" on page 116
- "Example" on page 117

Environment

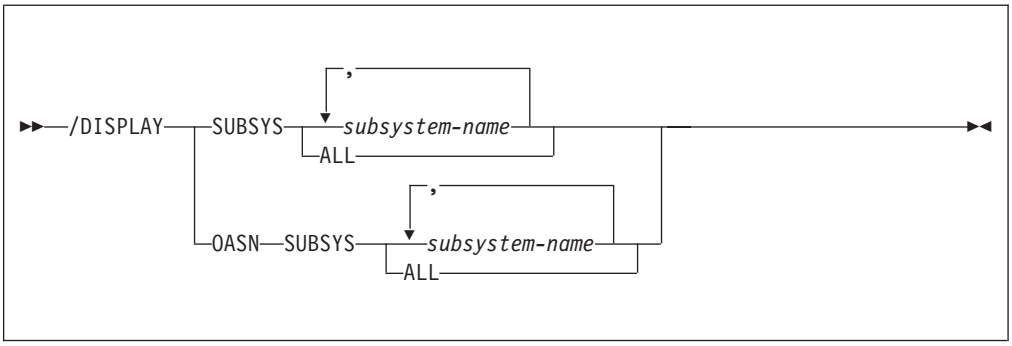
This command can be issued only from an IMS terminal.

Data sharing scope: Member

Authorization

This command requires an appropriate level of IMS authority, as described in *IMS Administration Guide: System*.

Syntax



Option descriptions

One of the following options is required:

SUBSYS
Identifies the subsystems to display information about.

subsystem-name, **...**
Specifies one or more subsystems. See "Output" on page 116 for a description of possible subsystem status.

© Copyright IBM Corp. 1983, 2008

ALL
Displays information about all subsystems.

OASN SUBSYS
Displays the outstanding recovery units (origin application schedule numbers, or OASN) associated with the external subsystems. The OASN is assigned by IMS when it schedules an application into a dependent region. That, coupled with the IMS ID, becomes the recovery token for units of work distributed to other subsystems.

subsystem-name, **...**
Specifies one or more subsystems to display information about.

ALL
Displays the outstanding recovery units associated with all external subsystems.

Output

The command recognition character (CRC) is displayed for each external subsystem. Subsystem status is one of the following:

CONNECTED
An IMS control region or dependent region has successfully connected to the external subsystem. At this point, the two systems can begin a normal dialog.

NOT CONNECTED
The external subsystem is in an idle state. That is, either it has not been the object of the /START SUBSYS command, or the external subsystem initialization exit routine indicated not to connect.

CONNECT IN PROGRESS
The connection process for the specified subsystem is in progress.

STOPPED
The specified subsystem has been stopped with the /STOP SUBSYS command. All region connections to the specified external subsystem have been terminated.

STOP IN PROGRESS
The /STOP SUBSYS command is in progress. Before it completes successfully, all active connections to the specified subsystem from all IMS regions must be quiesced.

INVALID SUBSYSTEM NAME = *subsystem-name*
The indicated subsystem name has not been defined to the IMS subsystem PROCLIB member. Add the subsystem definition to the subsystem member and issue the /START SUBSYS command.

SUBSYSTEM *subsystem-name* **NOT DEFINED BUT RECOVERY OUTSTANDING**
The indicated subsystem name has not been defined to IMS in the external subsystem PROCLIB member, but IMS still has outstanding recovery elements from a previous execution when the name was known. To resolve the recovery element problem, either add the indicated subsystem definition to the external subsystem PROCLIB member and then issue the /START SUBSYS command, or issue the /DISPLAY OASN SUBSYS command to determine the identification of the OASNs and then manually resolve the recovery elements by issuing the /CHANGE SUBSYS RESET command.

TERM IN PROGRESS
An internal termination of the subsystem is underway. This type of termination was instigated by IMS abnormal condition processing, an external subsystem exit, or the external subsystem.

A thread between an IMS dependent region and an external subsystem is created when an application program in the region establishes a connection to the external subsystem. The status of threads to an external subsystem is listed under the status of the subsystem. The absence of a list of threads under a connected subsystem indicates that no threads to the specified subsystem have been established.

Thread status can be one of the following:

CONNECTED(CONN)
An IMS control region or dependent region has successfully connected to the external subsystem.

ACTIVE
An IMS application program has established communication with an external subsystem.

The absence of a PSB name for a thread indicates that a connection to the external subsystem exists, but an application program is not currently occupying the region. The presence or absence of an LTERM name indicates whether a region is message-driven.

The preceding description of the /DISPLAY command is a partial description only. For a complete description, see *IMS Command Reference*.

Example

Display the status of connections between IMS and all subsystems.

```
/DISPLAY SUBSYSTEM ALL

SUBSYS    CRC    REGID    PROGRAM    LTERM    STATUS
SSTR      ?                                   CONN
                    1      DDLTLM17   PTERM01  CONN,ACTIVE
                    2      DDLTLM06   PTERM02  CONN
*85202/065933*
```

Chapter 20. -DISPLAY ARCHIVE (DB2)

The DB2 command DISPLAY ARCHIVE displays input archive log information.

Abbreviation: -DIS ARC

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Usage note"
- "Example" on page 120

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- ARCHIVE system privilege
- DISPLAY system privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax

Usage note

Data sharing members: Although the command ARCHIVE LOG SCOPE(GROUP) or ARCHIVE LOG MODE(QUIESCE) initiates archive processing for all members of a data sharing group, the command DISPLAY ARCHIVE shows information only for the member for which it is issued. To display input archive log information for all members of a data sharing group, enter the command on each member.

© Copyright IBM Corp. 1983, 2008

Example

Use the following command to display information about archive log data sets in use.

```
-DISPLAY ARCHIVE
```

This command produces output similar to the following output:

```
DSNJ322I  -DISPLAY ARCHIVE REPORT FOLLOWS-
                COUNT                  TIME
              (TAPE UNITS)           (MIN,SEC)
DSNZPARM          2                     0,00
CURRENT           2                     0,00
==============================
ADDR STATUS CORR-ID  VOLSER DATASET_NAME
03B0  BUSY  SHEDDEN  A00001 DSNT2AR1.DT25.D04169.T1328583.A0012701
     RECALL 03RCRSC  MIGRAT DSNT2AR1.DT25.D04169.T1334426.A0012704
A99B  BUSY  14DRSTRT ARN690 DSNT2AR1.DT25.D04169.T1346176.A0012705
BDDD  BUSY  10LPLALR ARN738 DSNT2AR1.DT25.D04170.T1506437.A0012743
END OF DISPLAY ARCHIVE REPORT.
```

This example report shows:

- The subsystem parameter values for MAX RTU (COUNT) and DEALLC PERIOD TIME as recorded in the DSNZPxxx load module
- Current specifications for the COUNT and TIME parameters
- Availability status of allocated archive log data sets
- Volume and data set names that are associated with current archive log read requests

Chapter 21. -DISPLAY BUFFERPOOL (DB2)

The DB2 command DISPLAY BUFFERPOOL displays the current status for one or more active or inactive buffer pools.

Abbreviation: -DIS BPOOL

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 122
- "Option descriptions" on page 122
- "Output" on page 124
- "Examples" on page 127

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

Authorization

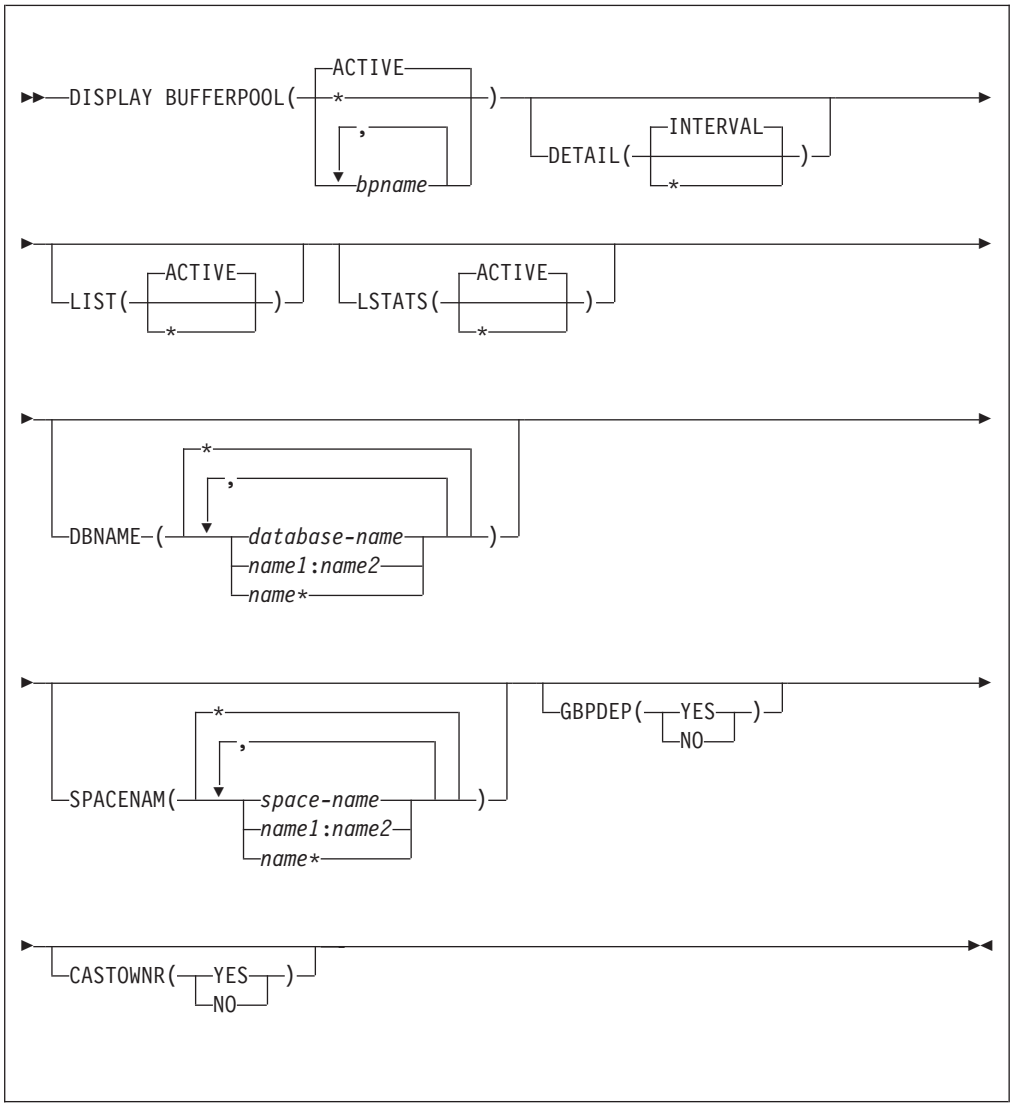
To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- DISPLAY system privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

(ACTIVE)
Displays the current buffer pool status for all active buffer pools.

(*) Displays the current buffer pool status for all active or inactive buffer pools.

(*bpname***)**
Names the buffer pool for which current status is to be displayed.
- 4-KB page buffer pools are named BP0, BP1, ..., BP49.
- 8-KB page buffer pools are named BP8K0, BP8K1, ..., BP8K9.
- 16-KB page buffer pools are named BP16K0, BP16K1, ..., BP16K9.
- 32-KB page buffer pools are named BP32K, BP32K1, ..., BP32K9.

DETAIL
Produces a detail report for one or more buffer pools. If DETAIL is not specified, a summary report is produced.

(INTERVAL)
Requests statistics accumulated since the last incremental display, or since the buffer pool was first activated if no previous incremental display exists.

(*) Requests statistics accumulated since the buffer pool was first activated.

LIST
Lists the open index spaces and table spaces associated with the buffer pools included in the report. Basic information is provided for non-data-sharing systems while more detail is provided if data sharing is active.

(ACTIVE)
Restricts the list of open index spaces and table spaces to those that are currently in use.

(*) Requests a list of all open index spaces and table spaces, whether currently in use or not.

LSTATS
Lists data set statistics for the open index spaces and table spaces associated with the buffer pools included in the report. The statistics displayed are incremental since the last time they were displayed.

(ACTIVE)
Restricts the list statistics to those data sets that are currently in use.

The **default** is **ACTIVE** when LIST is not specified or if LIST is specified with no parameter. If LIST is specified with a parameter and LSTATS has no parameter, the parameter specified for LIST is used for LSTATS.

(*) Includes statistics for all open index spaces and table spaces, whether currently in use or not.

DBNAME
Specifies which databases are included in the LIST display and the LSTATS display. If you specify DBNAME without LIST, LIST(ACTIVE) is assumed.

ABBREVIATION: DBN

(*database-name*, ...)
Identifies one or more databases to be included in the LIST and LSTATS displays. *database-name* can have any of the forms in the following list. In the list, *name1* and *name2* represent strings of one- to eight-characters. *name* represents a string of one- to eight-characters.

Form	Displays the status of...
name1	The database *name1*
name1:name2	All databases with names from *name1* to *name2* in a sorted list of database names.
*name**	All databases whose names begin with the string *name*

(*) Displays information on all databases that match the LIST specification. This is the default.

SPACENAM
Specifies which table spaces or index spaces within the specified databases to include in the LIST display and the LSTATS display. If you use SPACENAM without DBNAME, DBNAME(*) is assumed.

ABBREVIATION: SPACE

(*) Displays information about all table spaces and index spaces of the specified databases. This is the default.

(*space-name***, ...)**
Identifies one or more spaces to be included in the LIST and LSTATS displays. You can write *space-name* like *database-name* to designate:
- The name of a single table space or index space
- A range of names
- A partial name followed by a pattern-matching character

GBPDEP
Indicates whether to restrict the list of data sets to those that are group buffer pool dependent. This option is not valid if this is a non-data sharing DB2.

(YES)
Restricts the list of page sets to those that are group buffer pool dependent (GBP-dependent). An index space or table space is GBP-dependent if either of these conditions are true:
- Inter-DB2 R/W interest exists in it.
- Changed pages from it exist in the group buffer pool that have not yet been written to disk.

(NO)
Restricts the list of page sets to those that are non-group buffer pool dependent.

CASTOWNR
Indicates whether to restrict the list of data sets to those for which this DB2 member is the castout owner. This option is not valid if this is a non-data sharing DB2.

(YES)
Restricts the list of page sets for which this DB2 member is the castout owner.

(NO)
Restricts the list of page sets for which this DB2 member is not the castout owner.

Output

You can request a summary report or a detail report.

Summary report

A summary report contains the following information, as seen in "Example 1" on page 127:

Identification

BUFFERPOOL NAME
Buffer pool external name (BP0, BP1, ..., BP49, or BP32K, BP32K1, ..., BP32K9).

BUFFERPOOL ID
Buffer pool internal identifier (0-49, 80-89, 100-109, 120-129).

USE COUNT
Number of open table spaces or index spaces that reference this buffer pool. (Inactive pools have a zero use count.)

BUFFERPOOL SIZE
User-specified buffer pool size.

BUFFERS ALLOCATED
Number of allocated buffers in an active buffer pool.

TO BE DELETED
Number of buffers to be deleted in an active buffer pool (because of pool contraction).

IN-USE/UPDATED
Number of currently active (non-stealable) buffers in the buffer pool.

BUFFERS ACTIVE
Number of currently active (containing data) buffers in the buffer pool. For a fixed buffer pool, BUFFERS ACTIVE also indicates the number of buffers that are fixed in real storage.

Thresholds

VP SEQUENTIAL
Sequential steal threshold for the buffer pool.

DEFERRED WRITE
Deferred write threshold for the buffer pool.

VERTICAL DEFERRED WRT
Vertical deferred write threshold for the buffer pool.

PARALLEL SEQUENTIAL
Parallel sequential threshold for the buffer pool.

ASSISTING PARALLEL SEQT
Assisting parallel sequential threshold for the buffer pool.

Detail report

A detail report includes all summary report information and additional buffer pool related statistics. You can request cumulative statistics (accumulated since DB2 startup) or incremental statistics (accumulated since the last incremental display). A sample report appears in "Example 2" on page 127. The statistics in a detail report are grouped in the following categories:

Getpage information (message DSNB411I):

RANDOM GETPAGE
Non-sequential getpage requests

SYNC READ I/O(R)
Synchronous read I/O operations for non-sequential getpage

SEQ. GETPAGE
Sequential getpage requests

SYNC READ I/O(S)
Synchronous read I/O operations for sequential getpage

DMTH HIT
Number of times data management threshold reached

PAGE-INS REQ
Number of page-ins required for read I/O

Sequential prefetch statistics (message DSNB412I):

REQUESTS
Sequential prefetch requests

PREFETCH I/O
Sequential prefetch read I/O operations

PAGES READ
Number of pages read with sequential prefetch

List prefetch statistics (message DSNB413I):

REQUESTS
List prefetch requests

PREFETCH I/O
List prefetch read I/O operations

PAGES READ
Number of pages read due to list prefetch

Dynamic prefetch statistics (message DSNB414I):

REQUESTS
Dynamic prefetch requests

PREFETCH I/O
Dynamic prefetch read I/O operations

PAGES READ
Number of pages read with dynamic prefetch

Disabled prefetch statistics (message DSNB415I):

NO BUFFER
Prefetch disabled - no buffer

NO READ ENGINE
Prefetch disabled - no read processor

Page update statistics (message DSNB420I):

SYS PAGE UPDATES
System page updates

SYS PAGES WRITTEN
System pages written

ASYNC WRITE I/O
Asynchronous write I/O operations

SYNC WRITE I/O
Synchronous write I/O operations

PAGE-INS REQ
Number of page-ins required for read I/O

Page write statistics (message DSNB421I):

DWT HIT
Number of times deferred write threshold reached

VERTICAL DWT HIT
Number of times vertical deferred write threshold reached

NO WRITE ENGINE
Number of times write processor not available for I/O operations

Parallel processing activity (message DSNB440I):

PARALLEL REQUEST
Number of negotiations for task streams for parallel processing activity

DEGRADED PARALLEL
Number of times negotiation resulted in a degraded mode of operation

Examples

Example 1: A summary report is the default report if the DETAIL option is not specified. The following example shows a summary report that can be produced by the command:

```
-DISPLAY BUFFERPOOL(BP0) LIST(*) DBNAME(DSN8*)
```

```
DSNB401I - BUFFERPOOL NAME BP0, BUFFERPOOL ID 0, USE COUNT 20
DSNB402I - BUFFERPOOL SIZE             = 2000
           BUFFERS ALLOCATED           = 2000
           TO BE DELETED               = 0
           IN-USE/UPDATED              = 0
           BUFFERS ACTIVE              = 2000
DSNB404I - THRESHOLDS -
             VP SEQUENTIAL             = 80
             DEFERRED WRITE            = 85
             VERTICAL DEFERRED WRT     = 80
             PARALLEL SEQUENTIAL       = 50
             ASSISTING PARALLEL SEQT   = 0
DSNB406I -  PGFIX ATTRIBUTE -
             CURRENT                   = NO
             PENDING                   = YES
             PAGE STEALING METHOD      = LRU
DSN9022I - DSNB1CMD '-DISPLAY BUFFERPOOL' NORMAL COMPLETION
```

Example 2: A detail report can be generated that includes all summary report information and additional buffer pool related statistics. The following example shows a detail report that can be produced by the command:

```
-DISPLAY BUFFERPOOL(BP0) DETAIL
```

```
DSNB401I - BUFFERPOOL NAME BP0, BUFFERPOOL ID 0, USE COUNT 10
DSNB402I - BUFFERPOOL SIZE             = 2000
           BUFFERS ALLOCATED           = 2000
           TO BE DELETED               = 0
           IN-USE/UPDATED              = 200
#          BUFFERS ACTIVE              = 2000
DSNB404I - THRESHOLDS -
             VP SEQUENTIAL             = 80
             DEFERRED WRITE            = 50
             VERTICAL DEFERRED WRT     = 10
             PARALLEL SEQUENTIAL       = 50
             ASSISTING PARALLEL SEQT   =  0

DSNB409I - INCREMENTAL STATISTICS SINCE 10:32:48 OCT 23, 1993

DSNB411I - RANDOM GETPAGE    =      230 SYNC READ I/O (R) =      180
           SEQ.   GETPAGE    =      610 SYNC READ I/O (S) =       20
           DMTH HIT          =        0 PAGE-INS REQ      =       40

DSNB412I - SEQUENTIAL PREFETCH -
             REQUESTS        =        0   PREFETCH  I/O   =        0
             PAGES READ      =        0

DSNB413I - LIST PREFETCH -
             REQUESTS        =        0   PREFETCH  I/O   =        0
             PAGES READ      =        0

DSNB414I - DYNAMIC PREFETCH -
             REQUESTS        =        0   PREFETCH  I/O   =        0
             PAGES READ      =        0

DSNB415I - PREFETCH DISABLED -
             NO BUFFER       =        0   NO READ ENGINE  =           0
DSNB420I - SYS PAGE UPDATES  =        0 SYS PAGES WRITTEN =           0
           ASYNC WRITE I/O   =        0 SYNC WRITE I/O    =           0
           PAGE-INS REQ      =        0
DSNB421I - DWT HIT           =        0 VERTICAL DWT HIT  =           0
           NO WRITE ENGINE   =        0
|
DSNB440I - PARALLEL ACTIVITY -
             PARALL REQUEST  =        0   DEGRADED PARALL =       0

DSN9022I - DSNB1CMD '-DISPLAY BUFFERPOOL' NORMAL COMPLETION
```

Example 3: With the summary or detail report, you can list open table spaces and index spaces associated with the buffer pool. You can also request a display of statistics for each listed table space and index space. An example of a report generating this information could be produced by the command:

```
-DISPLAY BUFFERPOOL(BP0) LIST LSTATS
```

```
DSNB401I - BUFFERPOOL NAME BP0, BUFFERPOOL ID 0, USE COUNT 3
DSNB402I - BUFFERPOOL SIZE         = 2000
           BUFFERS ALLOCATED       = 2000
           TO BE DELETED           = 0
           IN-USE/UPDATED          = 200
           BUFFERS ACTIVE          = 2000
DSNB404I - THRESHOLDS -
             VP SEQUENTIAL         = 80
             DEFERRED WRITE        = 50
             VERTICAL DEFERRED WRT = 10
             PARALLEL SEQUENTIAL   = 50
             ASSISTING PARALLEL SEQT = 0
DSNB455I - SYNCHRONOUS I/O DELAYS -
             AVERAGE DELAY         = 22
             MAXIMUM DELAY         = 35
             TOTAL PAGES           = 23
DSN9022I - DSNB1CMD '-DISPLAY BUFFERPOOL' NORMAL COMPLETION
```

Chapter 22. -DISPLAY DATABASE (DB2)

The DB2 command DISPLAY DATABASE displays information about the status of the following objects:

- DB2 databases
- Table spaces
- Tables in segmented table spaces
- LOB table spaces
- Index spaces within a database
- Indexes on auxiliary tables
- Partitions of partitioned table spaces
- Partitions of index spaces

DISPLAY DATABASE RESTRICT indicates if a table space, index space, or partition is in any pending status. Use the ADVISORY option without the RESTRICT option to display any objects that are in an advisory pending status, such as the informational COPY-pending status or auxiliary warning advisory status.

In a data sharing environment, the command can be issued from any DB2 subsystem in the group that has access to the database.

Abbreviation: -DIS DB

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 133
- "Option descriptions" on page 134
- "Usage notes" on page 139
- "Output" on page 140
- "Examples" on page 141

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group

Authorization

The DISPLAY system privilege allows you to display status information for any database. The resulting display lists those databases for which the primary authorization ID or any of the secondary authorization IDs has the DISPLAYDB privilege. Error messages are produced for those databases specified over which the set of privileges does not include one of the following privileges or authorities:

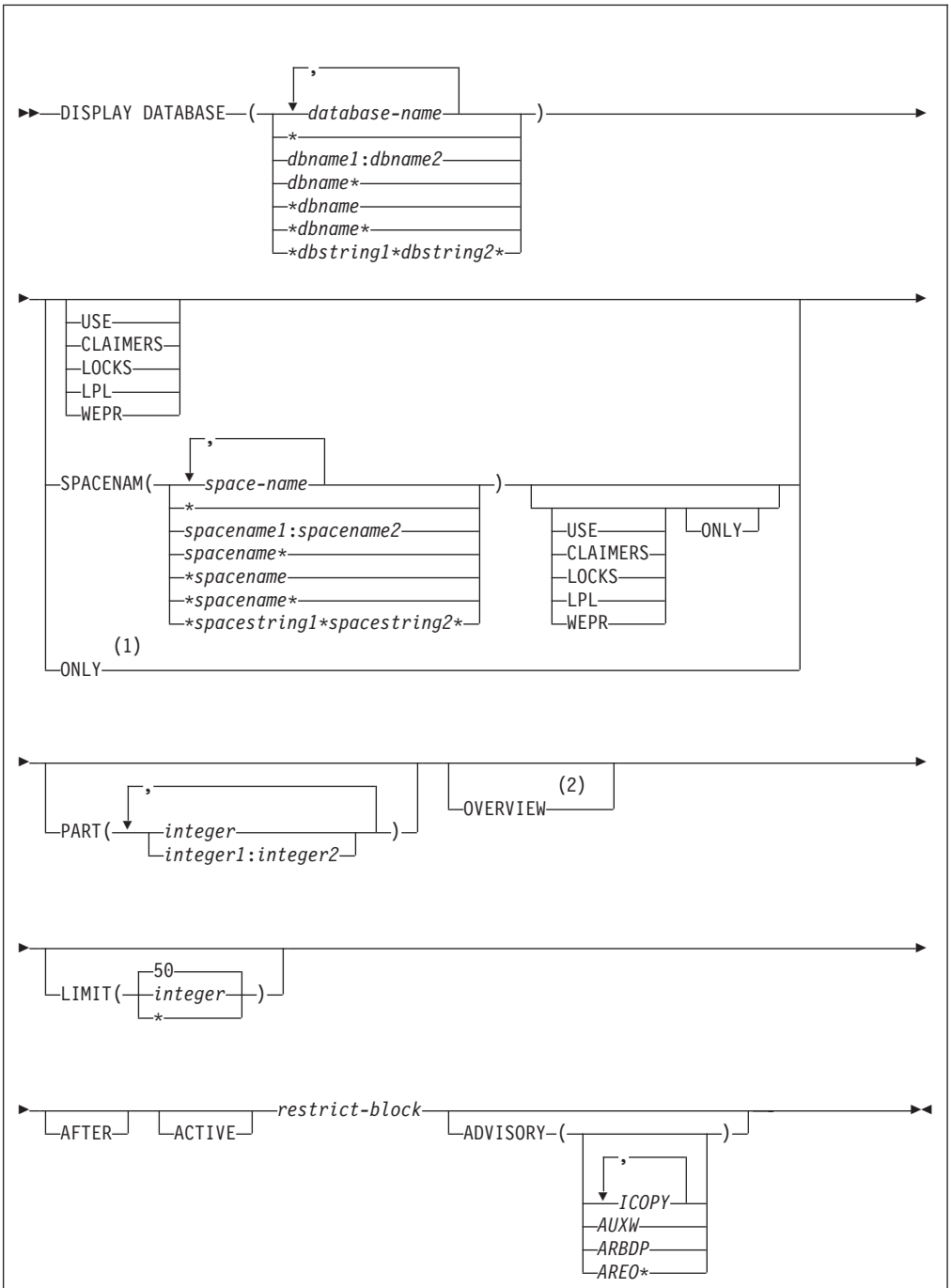
- DISPLAYDB privilege
- DISPLAY privilege

© Copyright IBM Corp. 1983, 2008

- DBMAINT authority
- DBCTRL authority
- DBADM authority
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax

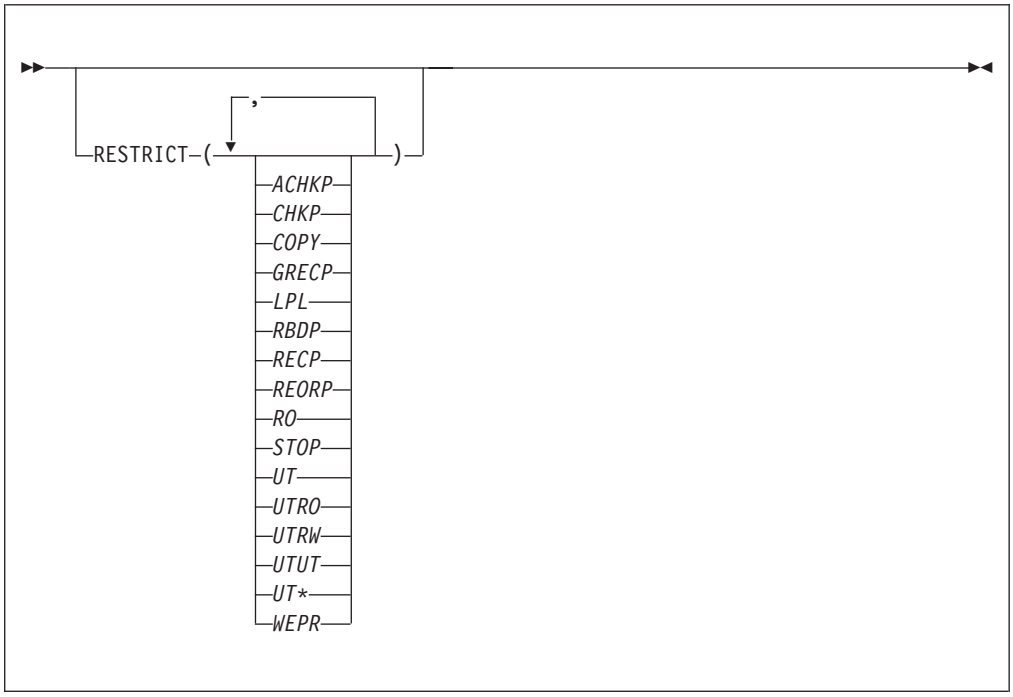


Notes:

1 If you specify the ONLY option without the SPACENAM() keyword, only the LIMIT, AFTER, and RESTRICT keywords apply.

2 The OVERVIEW keyword cannot be specified with any other keywords except SPACENAM, LIMIT, and AFTER.

restrict block:



Option descriptions

(*database-name***, ...)**
Identifies one or more databases whose status is to be displayed.

(*) Displays information on all databases that are defined to the DB2 subsystem for which the privilege set of the process has the required authorization.

dbname and *dbstring* can have any of the forms listed in Table 17 (where *dbname1* and *dbname2* represent any strings of from one to eight characters, and *dbname* represents any string of from one to seven characters).

Table 17. Forms of dbname and dbstring

Form	Displays the status of...
dbname1:dbname2	All databases whose names are between *dbname1* and *dbname2* inclusive
dbname*	All databases whose names begin with the string *dbname*
*dbname	All databases whose names end with the string *dbname*
dbname	All databases whose names contain the string *dbname*
*dbstring1*dbstring2*	All databases whose names contain the strings *dbstring1* and *dbstring2*

SPACENAM
Specifies what space to display. If you use SPACENAM, you must also specify the corresponding database name. If (*) is used to specify multiple databases, SPACENAM(*) can be specified to display all objects in these databases.

Abbreviation: SPACE, SP

(*space-name***, ...)**
Lists one or more spaces whose status is to be displayed. You can write *space-name* like *database-name* to designate:
- The name of a single table space or index space
- A range of names
- A partial name, including a beginning or ending pattern-matching character (*), a pattern-matching character between two strings, or any combination of these, with the following exception. Consecutive pattern-matching characters (*) are not allowed, and you cannot specify two pattern-matching characters in the middle of a keyword string.

(*) Displays information about all table spaces and index spaces of the specified database.

spacename and *spacestring* can have any of the forms listed in Table 18 (where *spacename1* and *spacename2* represent any strings of from one to eight characters, and *spacename* represents any string of from one to seven characters).

Table 18. Forms of spacename and spacestring

Form	Displays the status of...
spacename1:spacename2	All table spaces or index spaces whose names are between *spacename1* and *spacename2* inclusive
spacename*	All table spaces or index spaces whose names begin with the string *spacename*
*spacename	All table spaces or index spaces whose names end with the string *spacename*
spacename	All table spaces or index spaces whose names contain the string *spacename*
*spacestring1*spacestring2*	All table spaces or index spaces whose names contain the strings *spacestring1* and *spacestring2*

USE
Displays the following information:
- The applications and subsystems of the database or space that have claims or drains
- The applications and subsystems of the database or space on whose behalf locks for the space are held or waited for
- The connection IDs, correlation IDs, and authorization IDs for all applications allocated to spaces and partitions whose statuses are displayed
- The LUWID and location of any remote threads accessing the local database

CLAIMERS
Displays the following information:
- The claims on all table spaces, index spaces and partitions whose statuses are displayed
- The LUWID and location of any remote threads accessing the local database
- The connection IDs, correlation IDs, and authorization IDs for all applications allocated to spaces whose statuses are displayed
- The logical partitions that have logical claims and their associated claims

CLAIMERS overrides both LOCKS and USE. If you specify CLAIMERS, any references to LOCKS or USE are ignored.

LOCKS
Displays the following information:

- The applications and subsystems on whose behalf locks are held, waited on, or retained for the database or space
- The transaction locks for all table spaces, tables, index spaces and partitions whose statuses are displayed
- The connection IDs, correlation IDs, and authorization IDs for all applications allocated to spaces whose statuses are displayed
- The LUWID and location of any remote threads accessing the local database
- The drain locks for a resource held by running jobs
- The logical partitions that have drain locks and the drain locks that are associated with them
- The retained locks for a resource
- The page set or partition physical locks (P-locks) for a resource

LOCKS overrides USE. If both LOCKS and USE are specified, USE is ignored.

For a description of DB2 locking, see Part 5 (Volume 2) of *DB2 Administration Guide*.

LPL
Displays logical page list entries.

WEPR
Displays write error page range information.

ONLY
Displays information about the specified object.

without SPACENAM() keyword
Displays only database information. DB2 does not display information for the spaces within the database you specified with the DISPLAY DATABASE command. If you specify ONLY, the following keywords are valid:

- RESTRICT
- LIMIT
- AFTER

with SPACENAM() keyword
Displays the table spaces or indexes that have information requested by the DISPLAY DATABASE command. If you specify SPACENAM() ONLY, you must also specify one of the following keywords:

- USE
- CLAIMERS
- LOCKS
- LPL
- WEPR

DB2 displays tables with table locks when you specify both the LOCKS and ONLY keywords.

PART (*integer*, ...)
Indicates the partition number of one or more partitions whose status is to be displayed. The *integer* specified must identify a valid partition number for the corresponding space name and database name. *integer* can be written to designate one of the following values:

- A list of one or more partitions

- A range of all partition numbers that collate greater than or equal to *integer1* and less than or equal to *integer2*
- A combination of lists and ranges

OVERVIEW
Displays each object in the database on its own line, providing an easy way to see all objects in the database.

OVERVIEW limits the display to only the space names and space types that exist in the specified databases. The number of parts is displayed for any partitioned spaces.

The OVERVIEW keyword cannot be specified with any other keywords except SPACENAM, LIMIT, and AFTER.

LIMIT
Limits the number of messages to be displayed by the command.

(*integer***)**
Is the maximum number of messages that are to be displayed. The **default** is **50**. The maximum number of messages that can be displayed is limited by the space available.

(*) Limits the display to the space available.

AFTER
Displays the following information:

- If only a database name is used, AFTER continues the display of all other databases whose names collate greater than that name.
- If SPACENAM and a table space or index space name are used, AFTER continues the display to all other table spaces or index spaces in the same database whose names collate greater than that name.

AFTER cannot be used with more than one database name, table space name, index space name, with any pattern-matching character (*) within the database name, or with the **SPACENAM()** keyword.

ACTIVE
Limits the display to table spaces or index spaces that have had internal DB2 resources allocated to applications and are in a started state or to databases that contain such spaces.

Abbreviation: A

Default: Using neither ACTIVE nor RESTRICT displays information on all databases defined to DB2.

RESTRICT
Limits the display to databases, table spaces, or indexes in a restricted status. This includes those page sets that have logical page list entries. Specifying one or more keywords further limits the display to the named objects only.

Abbreviation: RES

Use of a database is restricted if the database is in any of the following situations:

- It is started for read-only processing.
- It is started for utility-only processing.
- It is stopped.

Use of a table space or index space is restricted if the table space or index space is in any of the following situations:

- It is in one of the three situations listed previously.

- It is being processed by a utility.
- It is in COPY-pending, CHECK-pending, RECOVER-pending, group buffer pool RECOVER-pending, auxiliary CHECK-pending, or REORG-pending status.
- It contains a page error range.
- It contains pages in the logical page list (LPL).

Specify one or more of the following keywords to limit objects that are to be listed.

ACHKP Displays objects in the auxiliary warning advisory state.

CHKP Display objects that are in CHECK-pending status.

COPY Display objects that are in COPY-pending status.

GRECP Displays objects that are in group buffer pool RECOVER-pending status.

LPL Displays logical page list entries.

RBDP Displays index objects that are in REBUILD- or RECOVER-pending status. This includes the restricted states RBDP, LPL, and WEPR.

RECP Displays objects that are in RECOVER-pending status, including the restricted states RECP, RECP*, LPL, and WEPR (write error page range).

REORP Displays objects that are in REORG-pending status.

RO Displays objects that are in read-only mode.

STOP Displays objects that are stopped, including the restricted states STOP, STOPE, STOPP, and LSTOP.

UT Displays objects that are in utility access mode.

UTRO Display objects that are serialized for utility access and available for read-only access.

UTRW Display objects that are serialized for utility access and available for read-write access.

UTUT Displays objects that are serialized for utility access and unavailable.

UT* Displays objects that are in any utility access mode: UT, UTRW, UTRO, or UTUT.

WEPR Displays write error page range information.

ADVISORY
Limits the display to indexes and table spaces to which read-write access is allowed, but for which some action is recommended.

Abbreviation: ADV

Use the DISPLAY DATABASE ADVISORY command without the RESTRICT option to determine when:

- An index space is in the informational COPY-pending (ICOPY) advisory status.
- A base table space or LOB table space is in the auxiliary warning (AUXW) advisory status.
- An index space is in the REBUILD-pending (ARBDP) advisory status.
- An index space is in the REORG (AREO*) advisory status.

Specify one or more of the following keywords to limit the objects listed.

AUXW
Displays objects that are in the auxiliary warning advisory state.

ICOPY
Displays objects that are in the informational COPY-pending advisory state.

ARBDP
Displays objects that are in the advisory REBUILD-pending status.

AREO*
Displays objects that are in the advisory REORG-pending status.

For information about resetting an advisory status, see Part 2 of *DB2 Utility Guide and Reference*.

Usage notes

Displaying DB2 catalog tables: You can always display the DB2 catalog tables. However, if a table space in the catalog containing information about user databases or user table spaces is stopped, those databases or table spaces cannot be displayed. Trying to display them will cause an error. See Appendix F of *DB2 SQL Reference* for a list of table space names and assigned tables.

If you issue DISPLAY DATABASE LOCKS on the catalog (DSNDB06), you might see a lock held on SYSDBASE with the correlation ID 020.DBCMD_05 or 020.DBCMD_06. This simply indicates the lock that DISPLAY DATABASE itself needs and is normal.

Displaying restricted and advisory status objects: To display all resources that are in restricted status, you must issue the DISPLAY DATABASE command twice. To display table spaces and indexes in restricted status, use the SPACENAM parameter with RESTRICT. To display databases in restricted status, do NOT use the SPACENAM parameter. Spaces could be unavailable even if they show RW mode if the database is in restricted status.

To display all resources that are in advisory status, issue the DISPLAY DATABASE ADVISORY command without the RESTRICT option. For information about resetting a restrictive or advisory status, see Part 2 of *DB2 Utility Guide and Reference*.

Communications Database and Resource Limit Facility: If the command specifies a table space or index space in the communications database or in the active resource limit facility database, the USE option displays the names of all members of the data sharing group that are using the specified table space or index space. Knowing which other members of the data sharing group might be using these spaces is useful when considering whether to drop table spaces and index spaces in the communications database and the resource limit facility database.

Displaying logical partitions: If you issue DISPLAY DATABASE with the PART parameter for a logical partition of a type 2 index, DB2 does not display physical claimers and physical locks in the output. Nonpartitioned indexes on partitioned tables are displayed with a type of 'IX' and with partition numbers displayed as 'L' followed by a four-digit number. When there is no information to be displayed at a logical partition level, partition numbers are displayed as 'L*'.

Displaying databases for declared temporary tables: DISPLAY DATABASE displays information about databases that are created with the AS TEMP option and the associated table spaces, but does not display information for declared temporary tables or index spaces that the database contains.

Displaying data-partitioned secondary indexes (DPSIs): DISPLAY DATABASE displays information about data-partitioned secondary indexes. DPSIs are displayed with a type of 'IX'. The partition number is displayed as 'D' followed by the four-digit partition number, ranging from 0001 to 4096.

Output

Message DSNT392I status information: The status codes that are displayed by the DISPLAY DATABASE command and their respective descriptions are as follows:

ARBDP
: Indicates that the index should be rebuilt to improve performance and allows the DB2 subsystem to pick this index for index-only access.

AREO*
: Indicates that the table space, index, or partition identified should be reorganized for optimal performance.

ACHKP
: Indicates an error in the LOB column of the base table space. The base table space has the auxiliary CHECK-pending restrictive status.

AREST
: Indicates that an object (a table space, index space, or a physical partition of a table space or index space) is in an advisory RESTART-pending state. If backout activity against the object is not already underway, initiate it either by issuing the RECOVER POSTPONED command, or by recycling the system with the system parameter LBACKOUT=AUTO.

AUXW
: Either the base table space is in the auxiliary warning advisory status, indicating an error in the LOB column, or the LOB table space is in the auxiliary warning advisory status, indicating an invalid LOB.

CHKP
: The object (a table space, a partition within a table space, or an index) is in the CHECK-pending status.

COPY
: The object (a table space or a partition within a table space) is in the COPY-pending status. An image copy is required for this object.

GRECP
: The object is GBP-dependent and a group buffer pool RECOVER is pending.

ICOPY
: The index space is in the informational COPY-pending advisory status.

LPL
: The object has entries in the logical page list.

LSTOP
: The logical partition of a nonpartitioning index is stopped.

PSRBD
: The entire nonpartitioning index space is in a page set REBUILD-pending status.

RBDP
: The physical or logical index partition is in the REBUILD-pending status.

RBDP* The logical partition of a nonpartitioning index is in the REBUILD-pending status, and the entire index is inaccessible to SQL applications. However, only the logical partition needs to be rebuilt.

RECP The object (a table space, table space partition, index space, index partition, or logical index partition) is in the RECOVER-pending status.

REFP The object (a table space, index space, or an index) is in the REFRESH-pending status.

RELDP
The object has a release dependency.

REORP
The data partition is in the REORG-pending status.

RESTP The table space or index space is in the restart-pending status.

RO The database, table space, table space partition, index space, or index space partition is started for read-only activity.

RW The database, table space, table space partition, index space, or index space partition is started for read and write activity.

STOP The database, table space, table space partition, index space, or index space partition is stopped.

STOPE
The table space or index space was implicitly stopped because there is a problem with the log RBA in a page. Message DSNT500I or DSNT501I is issued when the error is detected, indicating the inconsistency.

STOPP
A stop is pending for the database, table space, table space partition, index space, or index space partition.

UT The database, table space, table space partition, index space, or index space partition is started for utility processing only.

UTRO A utility is in process, on the table space, table space partition, index space, or index space partition, that allows only RO access. If the utility was canceled before the object was drained, the object can allow SQL access because the object was not altered by the utility.

UTRW A utility is in process, on the table space, table space partition, index space, or index space partition, that allows RW access.

UTUT A utility is in process, on the table space, table space partition, index space, or index space partition, that allows only UT access. If the utility was canceled before the object was drained, the object can allow SQL access because the object was not altered by the utility.

WEPR Displays write error page range information.

Examples

Example 1: Display information about table space TBS33 in database CB3. The USE option causes *connection-name*(CONNID), *correlation-id*(CORRID), and *authorization ID* (USERID) information to be displayed.

```
-DISPLAY DATABASE(CB3) SPACENAM(TBS33) USE
```

The following output is generated:

```
DSNT360I - ************************************************************
DSNT361I - *  DISPLAY DATABASE SUMMARY
             *    GLOBAL USE
DSNT360I - ************************************************************
DSNT362I -     DATABASE = CB3  STATUS = RW
               DBD LENGTH = 4028
DSNT397I -
NAME     TYPE PART  STATUS            CONNID   CORRID       USERID
-------- ---- ----- ----------------- -------- ------------ --------
TBS33    TS    0001 RW                LSS001   DSN2SQL      SYSADM
TBS33    TS    0002 RW                LSS001   DSN2SQL      SYSADM
TBS33    TS    0003 RW                LSS001   DSN2SQL      SYSADM
TBS33    TS    0004 RW                LSS001   DSN2SQL      SYSADM

******* DISPLAY OF DATABASE CB3      ENDED      **********************
DSN9022I . DSNTDDIS 'DISPLAY DATABASE' NORMAL COMPLETION
```

Example 2: Display information about table space TBS33 in database CB3. The LOCKS option displays lock information for table spaces and tables specified; LUWIDs and locations of any remote threads; and *connection-name*, *correlation-id*, and *authorization ID* information.

```
-DISPLAY DATABASE(CB3) SPACENAM(TBS33) LOCKS
```

The following output is generated:

```
DSNT360I - ************************************************************
DSNT361I - *  DISPLAY DATABASE SUMMARY
             *    GLOBAL LOCKS
DSNT360I - ************************************************************
DSNT362I -     DATABASE = CB3  STATUS = RW
               DBD LENGTH = 4028
DSNT397I -
NAME     TYPE PART  STATUS            CONNID   CORRID       LOCKINFO
-------- ---- ----- ----------------- -------- ------------ ---------
TBS33    TS    0001 RW
TBS33    TS    0002 RW
TBS33    TS    0003 RW
TBS33    TS    0004 RW                LSS004   DSN2SQL      H(IS,S,C)
TBS33    TS    0004 RW                LSS005   DSN2SQL      H(IS,S,C)

******* DISPLAY OF DATABASE CB3      ENDED      **********************
```

Example 3: Display information about table space TBS33 in database CB3. The CLAIMERS option displays claim types and durations; LUWIDs and locations of any remote threads; and *connection-name*, *correlation-id*, and *authorization ID* information.

```
-DISPLAY DATABASE(CB3) SPACENAM(TBS33) CLAIMERS
```

The following output is generated:

```
DSNT360I - ************************************************************
DSNT361I - *  DISPLAY DATABASE SUMMARY
             *    GLOBAL CLAIMERS
DSNT360I - ************************************************************
DSNT362I -     DATABASE = BP1DB  STATUS = RW
               DBD LENGTH = 4028
DSNT397I -
NAME     TYPE PART  STATUS            CONNID   CORRID      CLAIMINFO
-------- ---- ----- ----------------- -------- ------------ --------
BP1TS    TS    0001 RW,UTRW          BATCH    UPD1W        (WR,C)
BP1TS    TS    0001 RW,UTRW          BATCH    UPD1W        (CS,C)
BP1TS    TS    0001 RW,UTRW          UTILITY  REORG1       (CS,A)
BP1TS    TS    0002 RW,UTRW          UTILIT   REORG1       (CS,A)
BP1TS    TS    0003 RW,UTRW          UTILITY  REORG1       (CS,A)
BP1TS    TS    0004 RW,UTRW          UTILITY  REORG1       (CS,A)
```

```
#                   BP1TS    TS         RW               BATCH    UPD1W        (WR,C)
#                   BP1TS    TS         RW               BATCH    UPD1W        (CS,C)
#
#                   ******* DISPLAY OF DATABASE BP1DB      ENDED    *********************
```

Example 4: In a data sharing environment, display information about locks held for a table space. The application that is identified as LSS001 on member DB1G has locked partitions 1 and 2. LSS002 on member DB2G has locked partitions 1 and 3. Partition 4 has no locks held on it.

```
-DISPLAY DATABASE(DSN8D51A) SPACENAM(TSPART) LOCKS
```

The following output is generated:

```
NAME     TYPE PART  STATUS            CONNID   CORRID       LOCKINFO
-------- ---- ----- ----------------- -------- ------------ ---------
TSPART   TS    0001 RO                LSS001   DSN2SQL      H-IS,P,C
     -                    MEMBER NAME DB1G
TSPART   TS    0001 RO                                      H-S,PP,I
     -                    MEMBER NAME DB1G
TSPART   TS    0001 RO                LSS002   DSN2SQL      H-IS,P,C
     -                    MEMBER NAME DB2G
TSPART   TS    0001 RO                                      H-S,PP,I
     -                    MEMBER NAME DB2G
TSPART   TS    0002 RW                LSS001   DSN2SQL      H-IS,P,C
     -                    MEMBER NAME DB1G
TSPART   TS    0002 RW                                      H-S,PP,I
     -                    MEMBER NAME DB1G
TSPART   TS    0003 RW                LSS002   DSN2SQL      H-IS,P,C
     -                    MEMBER NAME DB2G
TSPART   TS    0003 RW                                      H-S,PP,I
     -                    MEMBER NAME DB2G
TSPART   TS    0004 RW
```

If DB2 cannot selectively lock the partitions, it must lock all of the partitions and the display looks similar to the following output. The LOCKINFO field shows a value of S, indicating that this is a table space lock. If partitions are held in different statuses, those statuses are listed below the table space locks.

```
NAME     TYPE PART  STATUS            CONNID   CORRID       LOCKINFO
-------- ---- ----- ----------------- -------- ------------ ---------
TSPART   TS                           LSS001   DSN2SQL      H-IS,S,C
     -                    MEMBER NAME DB1G
TSPART   TS                           LSS002   DSN2SQL      H-IS,S,C
     -                    MEMBER NAME DB2G
TSPART   TS    0001 RO                                      H-S,PP,I
     -                    MEMBER NAME DB1G
TSPART   TS    0002 RW                                      H-S,PP,I
     -                    MEMBER NAME DB2G
TSPART   TS    0003 RW                                      H-S,PP,I
     -                    MEMBER NAME DB2G
TSPART   TS    0004 RW
```

Example 5: Display information about page sets in database DSNDB01 that have entries in the logical page list. Limit the number of messages displayed to the space available.

```
-DB1G DISPLAY DATABASE(DSNDB01) SPACENAM(*) LIMIT(*) LPL
```

The following output is generated:

```
*********************************************************
DSNT361I -DB1G *  DISPLAY DATABASE SUMMARY
               *    GLOBAL LPL
DSNT360I -DB1G
*********************************************************
DSNT362I -DB1G    DATABASE = DSNDB01  STATUS = RW
```

```
               DBD LENGTH = 8000
DSNT397I -DB1G
NAME     TYPE PART  STATUS            LPL PAGES
-------- ---- ----- ----------------- ------------------
DBD01    TS         RW,LPL,GRECP      000001,000004,00000C,000010
----                                   000039-00003C
SPT01    TS         RW
SCT02    TS         RW
SYSLGRNG TS         RW
SYSUTILX TS         RW
SYSLGRNX TS         RW,LPL,GRECP      000000-FFFFFF
DSNSCT02 IX         RW
DSNSPT01 IX         RW
DSNSPT02 IX         RW
DSNLUX01 IX         RW
DSNLUX02 IX         RW
DSNLLX01 IX         RW
DSNLLX02 IX         RW
******* DISPLAY OF DATABASE DSNDB01  ENDED      **********************
DSN9022I -DB1G DSNTDDIS 'DISPLAY DATABASE' NORMAL COMPLETION
```

Example 6: Suppose DB2 is unable to selectively lock the partitions of table space TSPART, which is in database DSN8D81A. When you specify the following command, two applications are accessing TSPART, and the partitions have different statuses.

```
-DB1G DISPLAY DATABASE(DSN8D81A) SPACE(TSPART) PART(1,4) LOCKS
```

DB2 displays the locks as table space locks, as shown in the following output:

```
NAME     TYPE PART  STATUS            CONNID   CORRID       LOCKINFO
-------- ---- ----- ----------------- -------- ------------ ---------
TSPART   TS                          LSS001   DSN2SQL      H-IS,S,C
TSPART   TS                          LSS002   DSN2SQL      H-IS,S,C
TSPART   TS    0001 RO
TSPART   TS    0004 RW
```

Example 7: Suppose that you have executed the ALTER TABLESPACE statement on table space TSPART so that TSPART is now defined with LOCKPART YES. LOCKPART YES causes DB2 to do selective partition locking on TSPART. When you specify the following command, two applications are accessing TSPART. The application identified by connection ID LSS001 has locked partitions 1 and 2. The application identified by connection ID LSS002 has locked partitions 1 and 3.

```
-DB1G DISPLAY DATABASE(DSN8D81A) SPACE(TSPART) PART(1:4) LOCKS
```

DB2 displays the locks as partition locks, as shown in the following output:

```
NAME     TYPE PART  STATUS            CONNID   CORRID       LOCKINFO
-------- ---- ----- ----------------- -------- ------------ ---------
TSPART   TS    0001 RO                LSS001   DSN2SQL      H-IS,P,C
TSPART   TS    0001 RO                LSS002   DSN2SQL      H-IS,P,C
TSPART   TS    0002 RW                LSS001   DSN2SQL      H-IS,P,C
TSPART   TS    0003 RW                LSS002   DSN2SQL      H-IS,P,C
TSPART   TS    0004 RW
```

Example 8: Display information about all table spaces and index spaces in the range of databases from DBKD0101 to DBKD0106 that are in a restrictive status. Limit the number of messages that are displayed to the available space.

```
-DISPLAY DATABASE(DBKD0101,DBKD0103) SPACENAM(*) RESTRICT LIMIT(*)
```

The following output is generated:

```
DSNT360I - ***********************************
DSNT361I - *  DISPLAY DATABASE SUMMARY
             *    RESTRICTED
DSNT360I - ***********************************
DSNT362I -     DATABASE = DBKD0101  STATUS = RW
               DBD LENGTH = 4028
DSNT397I -
NAME     TYPE PART  STATUS            PHYERRLO PHYERRHI CATALOG  PIECE
-------- ---- ----- ----------------- -------- -------- -------- -----
TLKD0101 TS         RW,RESTP
IUKD011A IX         RW,RESTP
IXKD011B IX         RW,RESTP
```

Example 9: Display information about all table spaces that are in the auxiliary warning advisory status (AUXW), and all index spaces that are in informational COPY-pending status (ICOPY) in database DBIQUQ01. Limit the number of messages that are displayed to the available space.

```
-DISPLAY DATABASE(DBIQUQ01) SPACENAM(*) LIMIT(*) ADVISORY
```

The following output is generated:

```
DSNT360I - ***********************************
DSNT361I - *  DISPLAY DATABASE SUMMARY
             *    ADVISORY
DSNT360I - ***********************************
DSNT362I -     DATABASE = DBIQUQ01  STATUS = RW
               DBD LENGTH = 8066
DSNT397I -
NAME     TYPE PART  STATUS            PHYERRLO PHYERRHI CATALOG  PIECE
-------- ---- ----- ----------------- -------- -------- -------- -----
TPIQUQ01 TS    0001 RW,AUXW
     -THRU     0004
IAIQUQ01 IX         RW,ICOPY
IAIQUQ02 IX         RW,ICOPY
IAIQUQ03 IX         RW,ICOPY
IAIQUQ04 IX         RW,ICOPY
IPIQUQ01 IX    0001 RW,ICOPY
     -THRU     0004
IUIQUQ03 IX         RW,ICOPY
IXIQUQ02 IX         RW,ICOPY
******* DISPLAY OF DATABASE DBIQUQ01 ENDED      **********************
DSN9022I - DSNTDDIS 'DISPLAY DATABASE' NORMAL COMPLETION
```

Example 10: Display a list of all objects in database DB486A. This example shows five objects in the database. TS486A is a table space with four parts and TS486C is a nonpartitioned table space. IX486A is a nonpartitioned index for table space TS486A, IX486B is a partitioned index with four parts, and IX486C is a nonpartitioned index.

```
-DISPLAY DATABASE(DB486A) SPACE(*) OVERVIEW
```

The following output is generated:

```
DSNT360I - ***********************************
DSNT361I - *  DISPLAY DATABASE SUMMARY
             *    GLOBAL OVERVIEW
DSNT360I - ***********************************
DSNT362I -     DATABASE = DB486A  STATUS = RW
               DBD LENGTH = 4028
DSNT397I -
NAME     TYPE PART
-------- ---- -----
TS486A   TS    0004
IX486A   IX   L0004
IX486B   IX    0004
TS486C   TS
```

```
IX486C   IX
******* DISPLAY OF DATABASE DB486A   ENDED     **********************
DSN9022I - DSNTDDIS 'DISPLAY DATABASE' NORMAL COMPLETION
```

Example 11: Display database DB486B, specifying SPACE(*) for all spaces. This example shows table space TS486X with partitions 1 through 6 in STOP status. Partition 7 is in UT and COPY status, partition 8 is in STOP status, and partitions 9 and 10 are in RW status.

```
-DISPLAY DATABASE(DB486B) SPACE(*)
```

The following output is generated:

```
DSNT360I - ***********************************
DSNT361I - *  DISPLAY DATABASE SUMMARY
           *    GLOBAL OVERVIEW
DSNT360I - ***********************************
DSNT362I -     DATABASE = DB486B  STATUS = RW
               DBD LENGTH = 4028
DSNT397I -
NAME     TYPE PART  STATUS            PHYERRLO PHYERRHI CATALOG  PIECE
-------- ---- ----- ----------------- -------- -------- -------- -----
TS486X   TS    0001 STOP
    - THRU     0006
TS486X   TS    0007 UT,COPY
TS486X   TS    0008 STOP
TS486X   TS    0009 RW
    - THRU     0010
IX486X   IX   L0001 RW
IX486X   IX   L0002 LSTOP
    - THRU    L0003
IX486X   IX   L0004 LSTOP
    - THRU    L0010
IX486Y   IX    0001 RW
    - THRU     0010
IX486Z   IX         RW
******* DISPLAY OF DATABASE DB486B   ENDED     **********************
DSN9022I - DSNTDDIS 'DISPLAY DATABASE' NORMAL COMPLETION
```

Example 12: Display information about all indexes in the DBKD0101 database. INDEX2 contains information to be displayed at a logical level. Partitions 0001 and 0002 of INDEX3 are data-partitioned secondary indexes, as indicated by 'D' in the partition number.

```
-DISPLAY DATABASE(DBKD0101) SPACENAM(INDEX*)
```

The following output is generated:

```
DSNT360I - ***********************************
DSNT361I - *  DISPLAY DATABASE SUMMARY
           *    RESTRICTED
DSNT360I - ***********************************
DSNT362I -     DATABASE = DBKD0101  STATUS = RW
               DBD LENGTH = 4028
DSNT397I -
NAME     TYPE PART  STATUS            PHYERRLO PHYERRHI CATALOG  PIECE
-------- ---- ----- ----------------- -------- -------- -------- -----
INDEX1   IX    0001 RW
    -THRU      0002
INDEX2   IX   L*    RW
INDEX3   IX   D0001 RW
    -THRU     D0002
INDEX4   IX   L0001 RECP
INDEX4   IX   L0002 RW
```

Example 13: Display information about all table spaces in the DBKD0103 database that are in the advisory REBUILD-pending status (ARBDP) and the advisory REORG-status (AREO*). Limit the number of messages that are displayed to the available space. Assume that you specify the following command:

```
-DISPLAY DATABASE(DBKD0103) SPACENAM(*) LIMIT(*) ADVISORY(ARBDP,AREO*)
```

The following output is generated:

```
DSNT360I - **********************************
DSNT361I - *  DISPLAY DATABASE SUMMARY
           *    ADVISORY
DSNT360I - **********************************
DSNT362I -     DATABASE = DBKD0103  STATUS = RW
               DBD LENGTH = 16142
DSNT397I -
NAME     TYPE PART  STATUS            PHYERRLO PHYERRHI CATALOG  PIECE
-------- ---- ----- ----------------- -------- -------- -------- -----
PIX      IX    0001 RW,ARBDP,AREO*
    -THRU      0007
```

Example 14: Display information about table space DB2TSP in database DB2. The PART option includes both lists and ranges to display a very specific set of partitions. The table space underwent a single ROTATE operation before the final partitions were added.

```
-DISPLAY DATABASE(DB2) SPACENAME(DB2TSP) PART(1,2,4:6,9,10:12)
```

The following output is generated:

```
DSNT360I - **********************************
DSNT361I - *  DISPLAY DATABASE SUMMARY
           *    GLOBAL
DSNT360I - **********************************
DSNT362I -      DATABASE = DB2     STATUS = RW
                DBD LENGTH = 4028
DSNT397I -
NAME     TYPE PART  STATUS            PHYERRLO PHYERRHI CATALOG  PIECE
-------- ---- ----- ----------------- -------- -------- -------- -----
DB2TSP   TS    0002 RW,AREO*
DB2TSP   TS    0004 RW
DB2TSP   TS    0001 RW,AREO*
DB2TSP   TS    0005 RW
    -THRU      0006
DB2TSP   TS    0009 RW
    -THRU      0012
******* DISPLAY OF DATABASE DB2 ENDED     **************************
DSN9022I - DSNTDDIS 'DISPLAY DATABASE' NORMAL COMPLETION
```

Chapter 23. -DISPLAY DDF (DB2)

The DISPLAY DDF command displays information regarding the status and configuration of DDF, as well as statistical information regarding connections or threads controlled by DDF.

Abbreviation: -DIS DDF

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Output" on page 150
- "Examples" on page 150

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

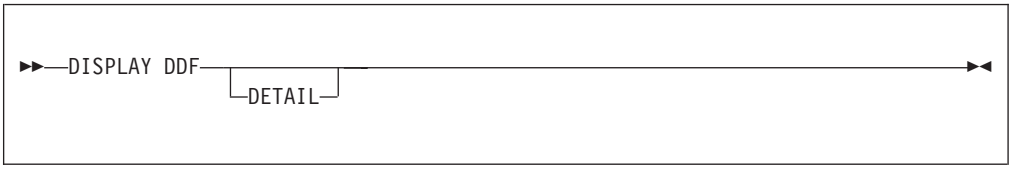
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- DISPLAY privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Option descriptions

DETAIL
Displays additional statistics and configuration information.

© Copyright IBM Corp. 1983, 2008

Output

The DISPLAY DDF command displays the following output:

STATUS The operational status of DDF.

LOCATION The location name of DDF.

LUNAME The fully qualified LUNAME of DDF.

GENERICLU The fully qualified generic LUNAME of DDF.

IPADDR The IP address of DDF.

TCPPORT The SQL listener port used by DDF.

RESPORT The resync listener port used by DDF.

DOMAIN The SQL and resync domains used by DDF.

Examples

Example 1: The following command is used to display a DDF detail report where DDF has not yet been started:

```
#display ddf detail
```

This command produces output similar to the following output:

```
DSNL080I  # DSNLTDDF DISPLAY DDF REPORT FOLLOWS:
DSNL081I STATUS=STOPDQ
DSNL082I LOCATION           LUNAME            GENERICLU
DSNL083I STL715B            -NONE.SYEC715B    -NONE
DSNL084I IPADDR          TCPPORT RESPORT
DSNL085I -NONE           447     5002
DSNL086I SQL    DOMAIN=-NONE
DSNL086I RESYNC DOMAIN=-NONE
DSNL090I DT=A  CONDBAT=     64 MDBAT=   64
DSNL092I ADBAT=    0 QUEDBAT=      0 INADBAT=      0 CONQUED=      0
DSNL093I DSCDBAT=      0 INACONN=      0
DSNL099I DSNLTDDF DISPLAY DDF REPORT COMPLETE
```

Example 2: The following command is used to display a DDF report, with no detail, where DDF is started:

```
#display ddf
```

This command produces output similar to the following output:

```
DSNL080I  # DSNLTDDF DISPLAY DDF REPORT FOLLOWS:
DSNL081I STATUS=STARTD
DSNL082I LOCATION           LUNAME            GENERICLU
DSNL083I STL715B            USIBMSY.SYEC715B  -NONE
DSNL084I IPADDR          TCPPORT RESPORT
DSNL085I 9.112.114.103   447     5002
DSNL086I SQL    DOMAIN=v7ec103.stl.ibm.com
DSNL086I RESYNC DOMAIN=v7ec103.stl.ibm.com
DSNL099I DSNLTDDF DISPLAY DDF REPORT COMPLETE
```

Example 3: The following command is used to display a DDF detail report where DDF is started:

```
#display ddf detail
```

This command produces output similar to the following output:

```
DSNL080I  # DSNLTDDF DISPLAY DDF REPORT FOLLOWS:
DSNL081I STATUS=STARTD
DSNL082I LOCATION           LUNAME            GENERICLU
DSNL083I STL715B            USIBMSY.SYEC715B  -NONE
DSNL084I IPADDR          TCPPORT RESPORT
DSNL085I 9.112.114.103   447     5002
DSNL086I SQL    DOMAIN=v7ec103.stl.ibm.com
DSNL086I RESYNC DOMAIN=v7ec103.stl.ibm.com
DSNL090I DT=A  CONDBAT=     64 MDBAT=   64
DSNL092I ADBAT=    1 QUEDBAT=      0 INADBAT=      0 CONQUED=      0
DSNL093I DSCDBAT=      0 INACONN=      0
DSNL099I DSNLTDDF DISPLAY DDF REPORT COMPLETE
```

Chapter 24. -DISPLAY FUNCTION SPECIFIC (DB2)

The DB2 command DISPLAY FUNCTION SPECIFIC displays statistics about external user-defined functions that DB2 applications access.

Abbreviation: -DIS FUNC SPEC

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 154
- "Option descriptions" on page 154
- "Usage notes" on page 154
- "Output" on page 155
- "Examples" on page 156

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group or local, depending on the value of the SCOPE option.

Authorization

To run this command, you must use a privilege set of the process that includes one of the following authorities for each function:

- Ownership of the function
- DISPLAY privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

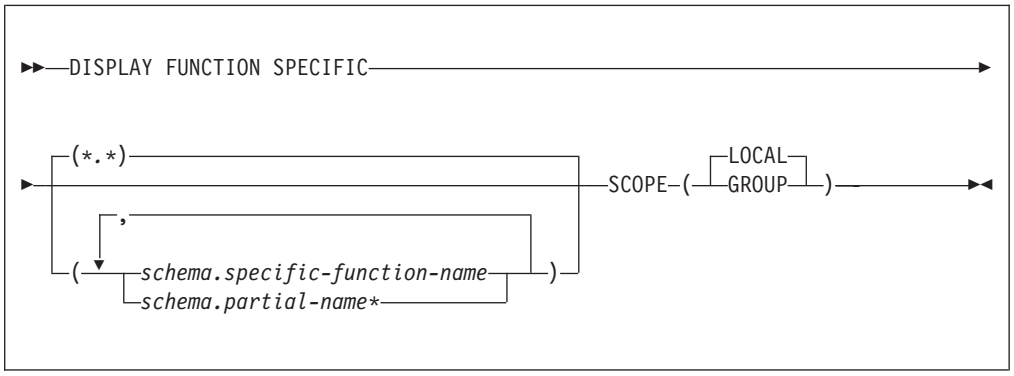
If you specify DISPLAY FUNCTION SPECIFIC *.* or *schema.partial-name**, you must use a privilege set of the process that includes one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

schema.specific-function-name
Displays information for the specific named function in the specified schema. You cannot specify a function name as you can in SQL; you must use the specific name. If a specific name was not specified on the CREATE FUNCTION statement, query SYSIBM.SYSROUTINES for the correct specific name:

```
SELECT SPECIFICNAME, PARM_COUNT
  FROM SYSIBM.SYSROUTINES
   WHERE NAME='function_name'
    AND SCHEMA='schema_name';
```

For overloaded functions, this query can return multiple rows.

*schema.partial-name**
Displays information for a set of functions in the specified schema.

The specific names of all functions in the set begin with *partial-name* and can end with any string, including the empty string. For example, schema1.ABC* displays information for all functions with specific names that begin with ABC in schema1.

(*.*)
Displays information for all functions that DB2 applications have accessed since the DB2 subsystem was started.

SCOPE
Specifies the scope of the command.

(LOCAL)
Specifies that the display includes information from only the local member.

(GROUP)
Specifies that the display includes information from all members of the data sharing group.

Usage notes

If you do not specify a partial or specific function name, DB2 displays information for all functions that DB2 applications have accessed since the DB2 subsystem was started.

This command does not apply to built-in functions or user-defined functions that are sourced on another function.

Output

This command displays one line of output for each function that a DB2 application has accessed.

Information returned by this command reflects a dynamic status. By the time DB2 displays the information, the status might have changed.

Sample output: The DISPLAY FUNCTION SPECIFIC command generates the following output:

```
DSNX975I = DSNX9DIS DISPLAY FUNCTION SPECIFIC REPORT FOLLOWS -

------- SCHEMA=PAYROLL
FUNCTION           STATUS ACTIVE QUED MAXQ TIME FAIL  WLM_ENV
APPL1              STARTED     1    0    0    0    0  PAYROLL
APPL2              STARTED     1    0    0    0    1  PAYROLL
APPL3              STARTED     0    1    2    0    0  PAYROLL
APPL5              STOPREJ     0    0    0    0    0  SANDBOX
APPL6              STOPABN     0    0    0    0    1  PAYROLL
FUNC1              STOPQUE     0    0    0    0    0  SANDBOX

DSNX9DIS DISPLAY FUNCTION SPECIFIC REPORT COMPLETE
DSN9022I = DSNX9COM '-DISPLAY FUNC' NORMAL COMPLETION
```

DISPLAY FUNCTION SPECIFIC command output: The DISPLAY FUNCTION SPECIFIC command displays the following output:

FUNCTION The specific name of the function.

STATUS The status of the function. The possible values are:

- STARTED — Requests for the function can be processed.
- STOPQUE — Requests are queued.
- STOPREJ — Requests are rejected.
- STOPABN — Requests are rejected because of abnormal termination.

ACTIVE The number of threads that are currently running the function.

QUED The number of threads that are waiting for the function to be scheduled.

MAXQ The maximum number of threads that have waited concurrently for the function to be scheduled since the DB2 subsystem was started.

TIME The number of times an SQL statement timed out while waiting for a request for the function to be scheduled.

FAIL The number of times a procedure has failed. DB2 resets this value to 0 each time you run the START FUNCTION command.

WLM_ENV The WLM environment where the function runs.

Message DSNX971I lists a range of functions that are stopped because a STOP FUNCTION SPECIFIC command included a partial name with the pattern-matching character (*). See Chapter 82, "-STOP FUNCTION SPECIFIC (DB2)," on page 409 for more information.

Examples

Example 1: Display information about functions in the PAYROLL schema and the HRPROD schema.

```
-DISPLAY FUNCTION SPECIFIC(PAYROLL.*, HRPROD.*)
```

This command produces output similar to the following output:

```
DSNX975I = DSNX9DIS DISPLAY FUNCTION SPECIFIC REPORT FOLLOWS-

------ SCHEMA=PAYROLL
FUNCTION        STATUS      ACTIVE   QUED   MAXQ   TIMEOUT   FAIL    WLM_ENV
PAYRFNC1        STARTED       0       0      1       0       0      WLMENV1
PAYRFNC2        STOPQUE       0       5      5       3       0      WLMENV1
PAYRFNC3        STARTED       2       0      6       0       0      WLMENV1
USERFNC4        STOPREJ       0       0      1       0       0      WLMENV3

------ SCHEMA=HRPROD
FUNCTION        STATUS      ACTIVE   QUED   MAXQ   TIMEOUT   FAIL    WLM_ENV
HRFNC1          STARTED       0       0      1       0       0      WLMENV2
HRFNC2          STOPREJ       0       0      1       0       0      WLMENV2
DSNX9DIS DISPLAY FUNCTION SPECIFIC REPORT COMPLETE
DSN9022I = DSNX9COM '-DISPLAY FUNC' NORMAL COMPLETION
```

Example 2: Display information about specific functions in the PAYROLL schema.

```
-DISPLAY FUNCTION SPECIFIC(PAYROLL.USERFNC2,PAYROLL.USERFNC4)
```

This command produces output similar to the following output:

```
DSNX975I = DSNX9DIS DISPLAY FUNCTION SPECIFIC REPORT FOLLOWS-

------ SCHEMA=PAYROLL
FUNCTION        STATUS      ACTIVE   QUED   MAXQ   TIMEOUT   FAIL    WLM_ENV
USERFNC2        STOPQUE       0       5      5       3       0      WLMENV3
USERFNC4        STOPREJ       0       0      1       0       0      WLMENV3
DSNX9DIS DISPLAY FUNCTION SPECIFIC REPORT COMPLETE
DSN9022I = DSNX9COM '-DISPLAY FUNC' NORMAL COMPLETION
```

Example 3: Display information about all functions that are in the SYSADM schema that DB2 applications have accessed. Assume that the STOP FUNCTION SPECIFIC(SYSADM.FN*) ACTION(QUEUE) command is in effect at the time you Enter the following command:

```
-DISPLAY FUNCTION SPECIFIC(SYSADM.*)
```

This command produces output similar to the following output:

```
DSNX975I = DSNX9DIS DISPLAY FUNCTION SPECIFIC REPORT FOLLOWS-

------ SCHEMA=SYSADM
FUNCTION       STATUS    ACTIVE   QUED  MAXQ  TIMEOUT  FAIL   WLM_ENV
FNC1           STOPQUE     0       0     0       0      0     WLMENV1
FNC2           STOPQUE     0       0     0       0      0     WLMENV3
DSNX9DIS FUNCTIONS FN - FN* STOP QUEUE
DSNX9DIS DISPLAY FUNCTION SPECIFIC REPORT COMPLETE
DSN9022I = DSNX9COM '-DISPLAY FUNC' NORMAL COMPLETION
```

Chapter 25. -DISPLAY GROUP (DB2)

The DB2 command DISPLAY GROUP displays information about the data sharing group to which a DB2 subsystem belongs. DISPLAY GROUP DETAIL displays the DB2 subsystem and group mode (compatibility mode, enabling new function mode, or DB2 Version 8 new-function mode or later).

Abbreviation: -DIS GROUP

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions" on page 158
- "Usage notes" on page 158
- "Output" on page 158
- "Examples" on page 160

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group

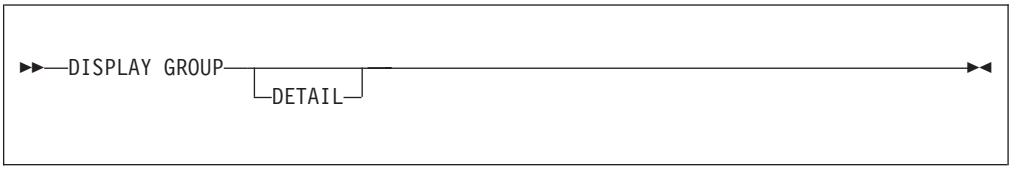
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- DISPLAY privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



© Copyright IBM Corp. 1983, 2008

Option descriptions

DETAIL
Displays information about the parallelism coordinator and parallelism assistant. If the DB2 data sharing group is in enabling new-function mode, DISPLAY GROUP DETAIL lists catalog and directory table spaces along with their status.

Usage notes

Member status: Message DSN7106I includes information about the XCF status of the members (STATUS in the display output). The status can be ACTIVE, QUIESCED, or FAILED.

ACTIVE indicates that the DB2 subsystem is active, and FAILED indicates that it is failed. A QUIESCED status results from a STOP DB2 command and consists of several subcategories:

QUIESCED
Indicates a normal quiesced state, as the result of a normal STOP DB2 command.

Q Q (quiesced) can be paired with one or more of the following letters:

I Indoubt or postponed abort units of recovery (URs) are outstanding. This means that retained locks are held.

C The member was shut down without completing castout processing. This event might have occurred because DB2 was started with the LIGHT(YES) option or stopped with the CASTOUT(NO) option, or because an error was encountered while attempting to cast out data from the coupling facility to the disk.

If DB2 encountered errors during castout processing, ensure that no connectivity problems exist between the coupling facility and the processor before restarting DB2.

R Retained information is needed for DB2 to perform resynchronization with one or more remote locations.

When DB2 is restarted, this resynchronization occurs.

ACTIVE
Indicates a normal active state without conditions.

A The member is active, but with the additional conditions. A (active) can be paired with the following letter:

I Indoubt or postponed abort units of recovery (URs) are outstanding. This indicates that retained locks are held.

Using this command in a non-data-sharing environment: DB2 issues the same response, except for information which does not exist: group name, member name, and member ID.

Output

DISPLAY GROUP command output: The DISPLAY GROUP command displays the following output:

*** BEGIN
The name of the DB2 group

DB2 MEMBER
The names of the DB2 group members

ID The IDs of the DB2 group members

SUBSYS
The subsystem names of the DB2 group members

CMDPREF
The command prefix for each member

STATUS
The status of each member (ACTIVE, QUIESCED with or without additional conditions, or FAILED)

SYSTEM NAME
The names of the z/OS system where the member is running, or was last running in cases when the member status is QUIESCED or FAILED

LVL A string of three numeric characters that list the following information:
- DB2 version
- DB2 release
- DB2 modification level

IRLM SUBSYS
The name of the IRLM subsystem to which the DB2 member is connected

IRLMPROC
The procedure names of the connected IRLM

SCA The SCA structure size in KB and the percentage currently in use

LOCK1
The LOCK1 structure size in KB.

The display also shows the following lock information:
- The maximum number of lock entries possible for the lock table and how many of those lock entries are currently in use. This number is an approximate value.
- The maximum number of modify lock list entries and how many of those list entries are currently in use.

For more information about the lock table and the list of modify locks, see Chapter 6 of *DB2 Data Sharing: Planning and Administration*.

PARALLELISM COORDINATOR
Indicates whether this DB2 member can coordinate parallel processing.

PARALLELISM ASSISTANT
Indicates whether this DB2 member can assist with parallel processing.

If the output indicates that either the lock structure or SCA are 0 % in use, it does not mean that the structure is empty. It could mean that the structure is very large and that the number of locks held or the number of records in the SCA is less than 1 %.

Description of message DSN7101I:

GROUP
The name of the data sharing group

GROUP LEVEL
A string of three numeric characters that lists the following information:
- DB2 version
- DB2 release
- DB2 modification level

DB2 release indicates the highest release with which any DB2 subsystem in the data sharing group has been started.

MODE
Indicates the catalog mode, as follows:
- C indicates compatibility mode.
- E indicates enabling new function mode.
- N indicates DB2 Version 8 new-function mode or later.

MODE is displayed only when you specify the DETAIL option.

PROTOCOL LEVEL
This value is supplied for serviceability only.

Examples

Example 1: The following sample output for a data sharing group can be generated by using the DISPLAY GROUP command:

```
-DB1A DISPLAY GROUP
```

```
DSN7100I -DB1A DSN7GCMD
*** BEGIN DISPLAY OF GROUP(DSNDB10 ) GROUP LEVEL(710)
           PROTOCOL LEVEL(2)  GROUP ATTACH NAME(DB10)
------------------------------------------------------------------
DB2                                    DB2 SYSTEM    IRLM
MEMBER   ID  SUBSYS CMDPREF   STATUS   LVL NAME      SUBSYS IRLMPROC
-------- --- ----   --------  -------- --- --------  ----   --------
DB1A       1 DB1A   -DB1A     ACTIVE   810 MVSA      DJ1A   DB1AIRLM
DB1B       2 DB1B   -DB1B     ACTIVE   810 MVSB      DJ1B   DB1BIRLM
DB1C       3 DB1C   -DB1C     ACTIVE   810 MVSC      DJ1C   DB1CIRLM
DB1D       4 DB1D   -DB1D     FAILED   710 MVSD      DJ1D   DB1DIRLM
DB1E       5 DB1E   -DB1E     QUIESCED 710 MVSE      DJ1E   DB1EIRLM
DB1F       6 DB1F   -DB1F     ACTIVE   810 MVSF      DJ1F   DB1FIRLM
DB1G       7 DB1G   -DB1G     ACTIVE   810 MVSG      DJ1G   DB1GIRLM
------------------------------------------------------------------
SCA   STRUCTURE SIZE:     1024 KB, STATUS= AC,   SCA IN USE:    11 %
LOCK1 STRUCTURE SIZE:     1536 KB
NUMBER  LOCK ENTRIES:      262144
NUMBER  LIST ENTRIES:        7353, LIST ENTRIES  IN USE:           0
*** END DISPLAY OF GROUP(DSNDB10 )
DSN9022I -DB1A DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

Example 2: In a non-data-sharing environment, the following sample output is generated by the DISPLAY GROUP command:

```
-DB1A DISPLAY GROUP
```

```
DSN7100I -DB1A DSN7GCMD
*** BEGIN DISPLAY OF GROUP(.......) GROUP LEVEL(...)
           PROTOCOL LEVEL(...)  GROUP ATTACH NAME(....)
-------------------------------------------------------------------------
DB2                                    DB2 SYSTEM    IRLM
MEMBER   ID  SUBSYS CMDPREF   STATUS   LVL NAME      SUBSYS IRLMPROC
-------- --- ----   --------  -------- --- --------  ----   --------
........   0 DB1A   -DB1A     ACTIVE   710 MVSA      DJ1A   DB1AIRLM
-------------------------------------------------------------------------
*** END DISPLAY OF GROUP(DSNDB10)
DSN9022I -DB1A DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

Example 3: You can obtain more information about the data sharing group, as shown in the following example, using the DISPLAY GROUP command with DETAIL option:

```
-DB1A DISPLAY GROUP DETAIL
```

```
DSN7100I -DB1A DSN7GCMD
*** BEGIN DISPLAY OF GROUP(DSNCAT1) GROUPLEVEL(810) MODE(E)
           PROTOCOL LEVEL(2)  GROUP ATTACH NAME(CAT1)
------------------------------------------------------------------------
DB2                                    SYSTEM        IRLM
MEMBER   ID  SUBSYS CMDPREF   STATUS   NAME     LVL  SUBSYS IRLMPROC
-------- --- ----   --------  -------- -------- ---  ----   --------
DB1A       1 DB1A   -DB1A     ACTIVE   MVSA     810  AR21   ARLM21
DB1B       2 DB1B   -DB1B     ACTIVE   MVSB     810  BR21   BRLM21
DB1C       3 DB1C   -DB1C     ACTIVE   MVSC     810  CRLM   CRLM21
DB2D       4 DB2D   -DB2D     FAILED   MVSD     810  DR21   DRLM21
DB2E       5 DB2E   -DB2E     QUIESCED MVSE     810  ER21   ERLM21
DB2F       6 DB2F   -DB2F     ACTIVE   MVSF     810  FR21   FRLM21
DB2G       7 DB2G   -DB2G     ACTIVE   MVSG     810  GR21   GRLM21
------------------------------------------------------------------------
DB2      PARALLEL    PARALLEL
MEMBER   COORDINATOR ASSISTANT
-------- ----------- ---------
DB2A             YES        NO
DB2B             YES       YES
DB2B             YES       YES
DB1C            ****      ****
DB2D            ****      ****
DB2E            ****      ****
DB2F               NO       YES
DB2G               NO        NO
------------------------------------------------------------------------
TABLE    ENABLED FOR
SPACE    NEW FUNCTION
-------- ------------
SPTR             YES
SYSDBASE         YES
SYSDBAUT         YES
SYSDDF           YES
SYSGPAUT         YES
SYSGROUP         YES
SYSGRTNS         YES
SYSHIST          YES
SYSJAVA          YES
SYSOBJ            NO
SYSPKAGE          NO
SYSPLANP          NO
SYSSEQ            NO
SYSSEQ2           NO
SYSSTATS          NO
SYSSTR            NO
SYSUSER           NO
SYSVIEWS          NO
------------------------------------------------------------------------

SCA   STRUCTURE SIZE:     1024 KB, STATUS= AC,   SCA IN USE:    11 %
LOCK1 STRUCTURE SIZE:     1536 KB,             LOCK1 IN USE: <   1 %
NUMBER  LOCK ENTRIES:      262144, LOCK ENTRIES  IN USE:          33
NUMBER  LIST ENTRIES:        7353, LIST ENTRIES  IN USE:           0
*** END DISPLAY OF GROUP(DSNCAT )
DSN9022I -DB1A DSN7GCMD 'DISPLAY GROUP ' NORMAL COMPLETION
```

Chapter 26. -DISPLAY GROUPBUFFERPOOL (DB2)

The DB2 command DISPLAY GROUPBUFFERPOOL displays information about the status of DB2 group buffer pools. It can also display related statistics.

Abbreviation: -DIS GBPOOL

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 164
- "Option descriptions" on page 164
- "Output" on page 165
- "Examples" on page 170

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group

Authorization

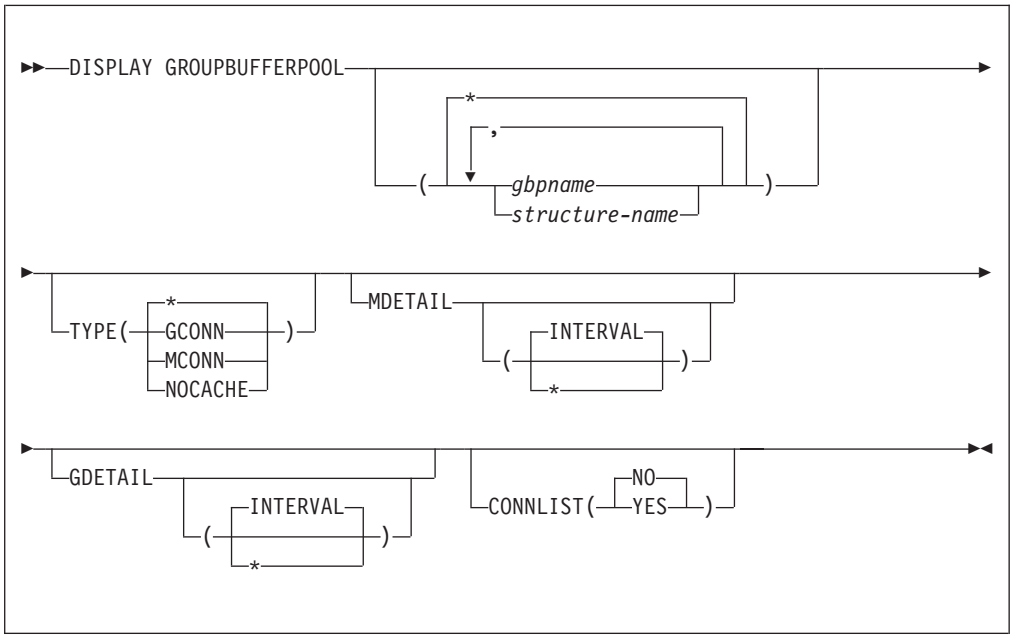
To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- DISPLAY privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

(*) Displays the group buffer pool status for all group buffer pools.

(*gbpname***)**
Names the group buffer pool for which status is to be displayed.
- 4-KB group buffer pools are named GBP0, GBP1, ..., GBP49.
- 8-KB group buffer pools are named GBP8K0, GBP8K1, ... , GBP8K9.
- 16-KB group buffer pools are named GBP16K0, GBP16K1, ... , GBP16K9.
- 32-KB group buffer pools are named GBP32K, GBP32K1, ... , GBP32K9.

(*structure-name***)**
Names the backing coupling facility structure for the group buffer pool. The coupling facility structure name has the following format:

```
groupname_gbpname
```

where *groupname* is the DB2 data sharing group name and the underscore (_) separates *groupname* and *gbpname*.

TYPE
Indicates the type of group buffer pools (among those that are specified) for which information is displayed.

(*) All group buffer pools are specified. This is the default.

(GCONN)
Group buffer pools that are currently connected to any member of the data sharing group. The connection can be “active” or “failed-persistent”.

(MCONN)
Group buffer pools that are currently connected to the member to which the command is directed.

(NOCACHE)
Group buffer pools that have the GBPCACHE attribute set to NO.

MDETAIL
Shows a detailed statistical report for the specified group buffer pools, reflecting the member's activity for each group buffer pool. If the member to which the command is directed has never been actively connected to the group buffer pool, no detail report is shown.

(INTERVAL)
Shows incremental statistics. The values displayed are accumulated since the last MDETAIL(INTERVAL) report for this member, if there was one. This is the default.

(*) Shows cumulative statistics. The values displayed are accumulated since this member first connected to the group buffer pool.

GDETAIL
Shows a detailed statistical report for the specified group buffer pools, reflecting the activity of the entire group for each group buffer pool. If the member to which the command is directed is not actively connected to the group buffer pool, no detail report is shown.

(INTERVAL)
Shows incremental statistics. The values displayed are accumulated since the last GDETAIL(INTERVAL) report, if there was one. This is the default.

(*) Shows cumulative statistics. The values displayed are accumulated since the group buffer pool was most recently allocated or re-allocated.

CONNLIST
Specifies whether a connection list report is shown for the specified group buffer pools, listing the connection names of the subsystems that are currently connected to the group buffer pools and their connection status.

(NO)
Do not show the connection list report.

(YES)
Show the connection list report.

Output

The DISPLAY GROUPBUFFERPOOL command can produce the following the three report types:
- A summary report
- A group detail report
- A member detail report

Summary report

You can display summary information about group buffer pools. The report indicates whether this DB2 subsystem is actively connected to the group buffer pools for which you requested information. The summary report also shows the following information:

Group buffer pool characteristics:
- Threshold values
- Directory-to-data entry ratio (both pending and current)
- Checkpoint interval
- Recovery status (whether damage assessment is pending)

CFRM policy information about the group buffer pool:

- The allocation value specified in the CFRM policy and whether the group buffer pool is currently allocated in the coupling facility.
- The actual allocated size (which can be different from that specified in the CFRM policy) and volatility status. DB2 requests non-volatile storage; however, it can allocate in a volatile structure.
- The actual number of directory entries, data pages, and connections to the group buffer pool.

The summary report contains additional information as follows:

AUTOMATIC RECOVERY
Indicates whether automatic recovery is allowed for this group buffer pool.

DUPLEX
Indicates the current duplexing option for the group buffer pool that is specified in the active CFRM policy.

REBUILD STATUS
Indicates whether a rebuild is in progress for this group buffer pool. If so, the phase of the rebuild is indicated as either QUIESCE, CONNECT, or CLEANUP. If the rebuild is in the process of stopping, the status indicates STOPPING.

DUPLEXING STATUS
Indicates the current state of the group buffer pool with respect to duplexing.

CFNAME
Indicates the name of the coupling facility in which the group buffer pool is allocated. If the group buffer pool is duplexed, this is the coupling facility name associated with the primary group buffer pool.

CFLEVEL
Indicates the level of the coupling facility in which the group buffer pool is allocated. If the group buffer pool is duplexed, this is the coupling facility level associated with the primary group buffer pool.

Both the operational and actual levels of the coupling facility are shown. The operational level indicates the capabilities of the coupling facility from the DB2 subsystem perspective. The actual level is the level as displayed by the z/OS D CF command.

LAST GROUP BUFFER POOL CHECKPOINT
Indicates the date and time of the last group buffer pool checkpoint, the LRSN that was recorded at that checkpoint, and the member name of the group buffer pool structure owner.

Group detail report

The group detail report shows detailed statistical information reflecting the activity of the entire group for the specified group buffer pools. This statistical information is helpful in tuning the size and other characteristics of group buffer pools. See Chapter 6 of *DB2 Data Sharing: Planning and Administration* for more information about using this information. The group detail report includes the same information as the summary report in addition to the following information:

READS
Information about reads.

This is a detailed accounting of the number of reads against the group buffer pool, including the following:

- The number of reads where data was returned.
- The number of reads where data was not returned, broken down to include more detailed information about whether the page was cached in the coupling facility or not, and whether directory entries needed to be created to fulfill requests for data.

WRITES
Information about writes.

This includes the number of writes for clean pages and changed pages, and how many writes failed because there was not enough storage in the group buffer pool.

CHANGED PAGES SNAPSHOT VALUE
The number of changed pages currently in the group buffer pool (a snapshot value).

RECLAIMS
The number of reclaims of directory entries and data pages.

CASTOUTS
The number of castouts.

CROSS INVALIDATIONS
The number of cross-invalidations that occurred because of directory reclaims and because of writes.

DUPLEXING STATISTICS FOR GBP*n*
This section of output indicates detailed duplexing statistics as follows:

CHANGED PAGES
Indicates the number of changed pages that are written to the secondary group buffer pool. If the group buffer pool has been duplexed for the entire reporting interval, this number approximates the CHANGED PAGES counter that is reported in message DSNB786I for the primary group buffer pool. The counts might not be exactly the same, due to timing periods for gathering the counter information for display or previous transaction failures that might have occurred.

FAILED DUE TO LACK OF STORAGE
Indicates the number of writes to the secondary group buffer pool that failed due to a lack of storage.

CHANGED PAGES SNAPSHOT VALUE
Indicates the number of changed pages that are currently cached in the secondary group buffer pool. This number approximates the CHANGED PAGES SNAPSHOT VALUE for the primary group buffer pool, but is probably not identical due to the asynchronous nature of gathering statistics for the two different coupling facility structures.

Member detail report

The member detail report includes the summary report and additional information about how a particular member's system is responding to the current environment. It categorizes reads and writes as synchronous or asynchronous. A large number of synchronous reads or writes can indicate that you need to tune your group buffer pools.

GBP CHECKPOINTS TRIGGERED
The number of checkpoints that occurred for this group buffer pool.

PARTICIPATION IN REBUILD
The number of times this member participated in a rebuild for this group buffer pool.

CASTOUTS
This section of output indicates detailed statistics for castout processing as follows:

PAGES CAST OUT
Indicates how many data pages were cast out of the group buffer pool by this member.

UNLOCK CASTOUT
The number of times that DB2 issued an unlock request to the coupling facility for castout I/Os that completed. As pages are cast out to disk, they are "locked for castout" in the coupling facility. The castout lock ensures that only one system is doing castout for a given page.

DB2 usually includes multiple pages in the write I/O request to disk for castout. Therefore, the UNLOCK CASTOUT counter should always be less than or equal to the value of the PAGES CASTOUT counter; it should be significantly less if multiple pages are written per I/O. For example, if there are four pages written per castout write I/O on average, PAGES CASTOUT should be four times larger than UNLOCK CASTOUT.

READ CASTOUT CLASS
Number of requests made to the group buffer pool to determine which pages belonging to a given page set or partition are cached in the group buffer pool as changed pages and thus need to be cast out.

READ CASTOUT CLASS is issued by the page set or partition castout owner, and it is also issued by the group buffer pool structure owner when the GBPOOLT threshold has been reached.

READ CASTOUT STATISTICS
The number of requests that are issued by the group buffer pool structure owner when the GBPOOLT threshold is reached. This determines which castout classes have changed pages. Generally READ CASTOUT STATISTICS is issued only once or twice for each occurrence of the GBPOOLT threshold.

READ DIRECTORY INFO
The number of requests to read the directory entries of all changed pages in the group buffer pool. The group buffer pool structure owner issues these requests at group buffer pool checkpoints. The purpose of the request is to determine the oldest recovery LRSN to use in case the group buffer pool fails. This recovery LRSN is displayed in message DSNB798I.

The request to read directory information might be issued several times for each group buffer pool checkpoint. If you see an abnormally high number here, it might be that the requests are being cut short by the model-dependent timeout criteria of the coupling facility. To help alleviate this problem, upgrade those coupling facilities to CFLEVEL=2 or above.

OTHER INTERACTIONS

This section of the output lists details of other interactions that this DB2 has with this group buffer pool.

REGISTER PAGE
: The number of times that DB2 registered interest to the group buffer pool for a single page. These are register-only requests, meaning that DB2 is not requesting that any data be returned for the page because no data is cached in the group buffer pool for this page. The REGISTER PAGE request is made only to create a directory entry for the page for cross-invalidation when downgrading the P-lock on a page set or partition from S mode to IS mode, or from SIX mode to IX mode.

UNREGISTER PAGE
: The number of times that DB2 reversed registered interest from the group buffer pool for a single page. This is generally done as DB2 uses pages from the local buffer pool that belong to partitions or page sets that are group buffer pool dependent.

DELETE NAME
: The number of times that DB2 issued a request to the group buffer pool to delete directory and data entries that were associated with a given page set or partition. DB2 issues this request:
  - When it converts a page set or partition from group buffer pool dependent to non group buffer pool dependent.
  - When the first DB2 member opens the object for GBPCACHE ALL objects.

READ STORAGE STATS
: The number of times that DB2 requested statistics information from the group buffer pool. This number should be relatively low. It is issued once per group buffer pool checkpoint by the group buffer pool structure owner. It is also issued for DISPLAY GROUPBUFFERPOOL GDETAIL requests and to record IFCID 0254.

DUPLEXING STATISTICS FOR GBP0-SEC

This section of the output lists details of other interactions that this DB2 has with this group buffer pool.

CHANGED PAGES
: Indicates the number of changed pages written to the secondary group buffer pool. This number approximates the sum of the synchronous writes of changed pages to the primary group buffer pool and the asynchronous writes of changed pages to the primary group buffer pool. The counts might not be exactly the same, due to timing periods for gathering the counter information for display or previous transaction failures that might have occurred.

FAILED DUE TO LACK OF STORAGE
: Indicates the number of writes to the secondary group buffer pool that failed due to a lack of storage.

COMPLETION CHECKS SUSPENDED
: Indicates the number of times DB2 checked for the completion of the write of a changed page to the secondary group buffer pool, but the write had not yet completed; DB2 suspends the execution unit until the write to the secondary group buffer pool completes.

DELETE NAME LIST
Indicates the number of DELETE NAME LIST requests to delete a set of pages from the secondary group buffer pool that have just been cast out to disk from the primary group buffer pool.

READ CASTOUT STATISTICS
Indicates the number of READ CASTOUT STATITICS requests to check for orphaned data entries in the secondary group buffer pool. The DB2 member that is the group buffer pool structure owner periodically issues these requests to determine whether garbage collection is necessary.

DELETE NAME
Indicates the number of DELETE NAME requests to delete orphaned data entries from the secondary group buffer pool. The DB2 member that is the group buffer pool structure owner issues these requests if it determines that garbage collection is necessary.

Examples

Example 1: This is an example of a summary report that can be produced by the following command:

```
-DISPLAY GROUPBUFFERPOOL(GBP29)
```

Message DSNB799I is displayed if the group buffer pool is duplexed and the secondary group buffer pool is currently allocated. If a secondary group buffer pool is not allocated, message DSNB799I is not included in the output.

```
DSNB750I - DISPLAY FOR GROUP BUFFER POOL GBP29 FOLLOWS
DSNB755I - DB2 GROUP BUFFER POOL STATUS
             CONNECTED                                = YES
             CURRENT DIRECTORY TO DATA RATIO          = 5
             PENDING DIRECTORY TO DATA RATIO          = 5
             CURRENT GBPCACHE ATTRIBUTE               = YES
             PENDING GBPCACHE ATTRIBUTE               = YES
DSNB756I -   CLASS CASTOUT THRESHOLD                  = 10%
             GROUP BUFFER POOL CASTOUT THRESHOLD       = 50%
             GROUP BUFFER POOL CHECKPOINT INTERVAL     = 8 MINUTES
             RECOVERY STATUS                          = NORMAL
             AUTOMATIC RECOVERY                       = Y
DSNB757I - MVS CFRM POLICY STATUS FOR DSNCAT_GBP29    = NORMAL
             MAX SIZE INDICATED IN POLICY             = 2048 KB
             DUPLEX INDICATOR IN POLICY               = ENABLED
             CURRENT DUPLEXING MODE                   = DUPLEX
             ALLOCATED                                = YES
DSNB758I -     ALLOCATED SIZE                         = 2048 KB
               VOLATILITY STATUS                      = VOLATILE
               REBUILD STATUS                         = DUPLEXED
               CFNAME                                 = CACHE01
               OPERATIONAL CFLEVEL                    = 5
               ACTUAL CFLEVEL                         = 7
DSNB759I -     NUMBER OF DIRECTORY ENTRIES            = 1950
               NUMBER OF DATA PAGES                   = 389
               NUMBER OF CONNECTIONS                  = 2
DSNB798I - LAST GROUP BUFFER POOL CHECKPOINT
                                      17:08:41 OCT 16, 2002
             GBP CHECKPOINT RECOVERY LRSN             = AF6BBAEF3307
             STRUCTURE OWNER                          = V61B
DSNB799I - SECONDARY GBP ATTRIBUTES
             ALLOCATED SIZE                           = 2048 KB
             VOLATILITY STATUS                        = VOLATILE
             CFNAME                                   = LF01
             OPERATIONAL CFLEVEL                      = 5
             ACTUAL CFLEVEL                           = 7
             NUMBER OF DIRECTORY ENTRIES              = 1950
             NUMBER OF DATA PAGES                     = 389
DSNB790I - DISPLAY FOR GROUP BUFFER POOL GBP29 IS COMPLETE
DSN9022I - DSNB1CMD '-DISPLAY GBPOOL' NORMAL COMPLETION
```

Example 2: Assume you want a summary report about group buffer pool 29 (GBP29), including all connections to that group buffer pool. Enter the following command:

```
-DISPLAY GROUPBUFFERPOOL(GBP29) CONNLIST(YES)
```

This command produces output similar to the following output:

```
DSNB750I - DISPLAY FOR GROUP BUFFER POOL GBP29 FOLLOWS
DSNB755I - DB2 GROUP BUFFER POOL STATUS
             CONNECTED                                = YES
             CURRENT DIRECTORY TO DATA RATIO          = 5
             PENDING DIRECTORY TO DATA RATIO          = 5
             CURRENT GBPCACHE ATTRIBUTE               = YES
             PENDING GBPCACHE ATTRIBUTE               = YES
DSNB756I -   CLASS CASTOUT THRESHOLD                  = 10%
             GROUP BUFFER POOL CASTOUT THRESHOLD       = 50%
             GROUP BUFFER POOL CHECKPOINT INTERVAL     = 8 MINUTES
             RECOVERY STATUS                          = NORMAL
             AUTOMATIC RECOVERY                       = Y
DSNB757I - MVS CFRM POLICY STATUS FOR DSNCAT_GBP29    = NORMAL
             MAX SIZE INDICATED IN POLICY             = 2048 KB
             DUPLEX INDICATOR IN POLICY               = ENABLED
             CURRENT DUPLEXING MODE                   = SIMPLEX
             ALLOCATED                                = YES
DSNB758I -     ALLOCATED SIZE                         = 2048 KB
               VOLATILITY STATUS                      = VOLATILE
               REBUILD STATUS                         = DUPLEXED
               CFNAME                                 = CACHE01
               OPERATIONAL CFLEVEL                    = 5
               ACTUAL CFLEVEL                         = 7
DSNB759I -     NUMBER OF DIRECTORY ENTRIES            = 1950
               NUMBER OF DATA PAGES                   = 389
               NUMBER OF CONNECTIONS                  = 2
DSNB798I - LAST GROUP BUFFER POOL CHECKPOINT
                                      17:08:41 OCT 16, 2002
             GBP CHECKPOINT RECOVERY LRSN             = AF6BBAEF3307
             STRUCTURE OWNER                          = V61B
DSNB799I - SECONDARY GBP ATTRIBUTES
             ALLOCATED SIZE                           = 2048 KB
             VOLATILITY STATUS                        = VOLATILE
             CFNAME                                   = LF01
             OPERATIONAL CFLEVEL                      = 5
             ACTUAL CFLEVEL                           = 7
             NUMBER OF DIRECTORY ENTRIES              = 1950
             NUMBER OF DATA PAGES                     = 389
DSNB766I - THE CONNLIST REPORT FOLLOWS
DSNB767I - CONNECTION NAME = DB2_V61B        , CONNECTION STATUS = D
               CONNECTOR'S RELEASE                  =     6100
DSNB767I - CONNECTION NAME = DB2_V61A        , CONNECTION STATUS = D
               CONNECTOR'S RELEASE                  =     6100
DSNB769I - THE CONNLIST REPORT IS COMPLETE
DSNB790I - DISPLAY FOR GROUP BUFFER POOL GBP29 IS COMPLETE
DSN9022I - DSNB1CMD '-DISPLAY GBPOOL' NORMAL COMPLETION
```

Example 3: This example shows a group detail report that is produced by the following command:

```
-DISPLAY GROUPBUFFERPOOL(GBP29) GDETAIL(*)
```

Message DSNB762I is displayed in the output only if the secondary group buffer pool is allocated.

```
DSNB750I - DISPLAY FOR GROUP BUFFER POOL GBP29 FOLLOWS
DSNB755I - DB2 GROUP BUFFER POOL STATUS
             CONNECTED                                = YES
             CURRENT DIRECTORY TO DATA RATIO          = 5
             PENDING DIRECTORY TO DATA RATIO          = 5
             CURRENT GBPCACHE ATTRIBUTE               = YES
             PENDING GBPCACHE ATTRIBUTE               = YES
DSNB756I -   CLASS CASTOUT THRESHOLD                  = 10%
             GROUP BUFFER POOL CASTOUT THRESHOLD       = 50%
             GROUP BUFFER POOL CHECKPOINT INTERVAL     = 8 MINUTES
             RECOVERY STATUS                          = NORMAL
             AUTOMATIC RECOVERY                       = Y
DSNB757I - MVS CFRM POLICY STATUS FOR DSNCAT_GBP29    = NORMAL
             MAX SIZE INDICATED IN POLICY             = 2048 KB
             DUPLEX INDICATOR IN POLICY               = ENABLED
             CURRENT DUPLEXING MODE                   = DUPLEX
             ALLOCATED                                = YES
DSNB758I -     ALLOCATED SIZE                         = 2048 KB
               VOLATILITY STATUS                      = VOLATILE
               REBUILD STATUS                         = DUPLEXED
               CFNAME                                 = CACHE01
               OPERATIONAL CFLEVEL                    = 5
               ACTUAL CFLEVEL                         = 7
DSNB759I -     NUMBER OF DIRECTORY ENTRIES            = 1950
               NUMBER OF DATA PAGES                   = 389
               NUMBER OF CONNECTIONS                  = 2
DSNB798I - LAST GROUP BUFFER POOL CHECKPOINT
                                        17:08:41 OCT 16, 2002
             GBP CHECKPOINT RECOVERY LRSN             = AF6BBAEF3307
             STRUCTURE OWNER                          = V61B
DSNB799I - SECONDARY GBP ATTRIBUTES
             ALLOCATED SIZE                           = 2048 KB
             VOLATILITY STATUS                        = VOLATILE
             CFNAME                                   = LF01
             OPERATIONAL CFLEVEL                      = 5
             ACTUAL CFLEVEL                           = 7
             NUMBER OF DIRECTORY ENTRIES              = 1950
             NUMBER OF DATA PAGES                     = 389
DSNB783I - CUMULATIVE GROUP DETAIL STATISTICS SINCE 17:08:35 OCT 16,
2002
DSNB784I - GROUP DETAIL STATISTICS
             READS
               DATA RETURNED                          = 4
DSNB785I -     DATA NOT RETURNED
                 DIRECTORY ENTRY EXISTED              = 0
                 DIRECTORY ENTRY CREATED              = 45
                 DIRECTORY ENTRY NOT CREATED          = 0, 0
DSNB786I -   WRITES
               CHANGED PAGES                          = 5
               CLEAN PAGES                            = 0
               FAILED DUE TO LACK OF STORAGE          = 0
             CHANGED PAGES SNAPSHOT VALUE             = 5
DSNB787I -   RECLAIMS
               FOR DIRECTORY ENTRIES                  = 0
               FOR DATA ENTRIES                       = 0
             CASTOUTS                                 = 0



DSNB788I -   CROSS INVALIDATIONS
               DUE TO DIRECTORY RECLAIMS              = 0
               DUE TO WRITES                          = 0
               EXPLICIT                               = 0
DSNB762I - DUPLEXING STATISTICS FOR GBP29-SEC
             WRITES
               CHANGED PAGES                          = 5
               FAILED DUE TO LACK OF STORAGE          = 0
             CHANGED PAGES SNAPSHOT VALUE             = 5
DSNB790I - DISPLAY FOR GROUP BUFFER POOL GBP29 IS COMPLETE
DSN9022I - DSNB1CMD '-DISPLAY GBPOOL' NORMAL COMPLETION
```

Example 4: This example shows the member detail section from the report that is produced by the following command:

```
-DISPLAY GROUPBUFFERPOOL(GBP29) MDETAIL(*)
```

Messages DSNB764I and DSNB793I are displayed in the output only if the secondary group buffer pool is allocated.

```
DSNB750I - DISPLAY FOR GROUP BUFFER POOL GBP29 FOLLOWS
DSNB755I - DB2 GROUP BUFFER POOL STATUS
             CONNECTED                                = YES
             CURRENT DIRECTORY TO DATA RATIO          = 5
             PENDING DIRECTORY TO DATA RATIO          = 5
             CURRENT GBPCACHE ATTRIBUTE               = YES
             PENDING GBPCACHE ATTRIBUTE               = YES
DSNB756I -   CLASS CASTOUT THRESHOLD                  = 10%
             GROUP BUFFER POOL CASTOUT THRESHOLD       = 50%
             GROUP BUFFER POOL CHECKPOINT INTERVAL     = 8 MINUTES
             RECOVERY STATUS                          = NORMAL
             AUTOMATIC RECOVERY                       = Y
DSNB757I - MVS CFRM POLICY STATUS FOR DSNCAT_GBP29     = NORMAL
             MAX SIZE INDICATED IN POLICY             = 2048 KB
             DUPLEX INDICATOR IN POLICY               = ENABLED
             CURRENT DUPLEXING MODE                   = DUPLEX
             ALLOCATED                                = YES
DSNB758I -     ALLOCATED SIZE                         = 2048 KB
               VOLATILITY STATUS                      = VOLATILE
               REBUILD STATUS                         = DUPLEXED
               CFNAME                                 = CACHE01
               OPERATIONAL CFLEVEL                    = 5
               ACTUAL CFLEVEL                         = 7
DSNB759I -     NUMBER OF DIRECTORY ENTRIES            = 1950
               NUMBER OF DATA PAGES                   = 389
               NUMBER OF CONNECTIONS                  = 2
DSNB798I - LAST GROUP BUFFER POOL CHECKPOINT
                                        17:08:41 OCT 16, 2002
             GBP CHECKPOINT RECOVERY LRSN             = AF6BBAEF3307
             STRUCTURE OWNER                          = V61B
DSNB799I - SECONDARY GBP ATTRIBUTES
             ALLOCATED SIZE                           = 2048 KB
             VOLATILITY STATUS                        = VOLATILE
             CFNAME                                   = LF01
             OPERATIONAL CFLEVEL                      = 5
             ACTUAL CFLEVEL                           = 7
             NUMBER OF DIRECTORY ENTRIES              = 1950
             NUMBER OF DATA PAGES                     = 389
DSNB772I - CUMULATIVE MEMBER DETAIL STATISTICS SINCE 17:08:41 OCT 16,
2002
DSNB773I - MEMBER DETAIL STATISTICS
             SYNCHRONOUS READS
               DUE TO BUFFER INVALIDATION
                 DATA RETURNED                        = 0
                 DATA NOT RETURNED                    = 0
DSNB774I -     DUE TO DATA PAGE NOT IN BUFFER POOL
                 DATA RETURNED                        = 0
                 DATA NOT RETURNED                    = 0
DSNB775I -   PREFETCH READS
               DATA NOT RETURNED                      = 0
DSNB789I -    REGISTER PAGE LIST                      = 0
               PAGES RETRIEVED                        = 0
          FAILED READS DUE TO LACK OF STORAGE         = 0
DSNB776I -   SYNCHRONOUS WRITES
               CHANGED PAGES                          = 5
               CLEAN PAGES                            = 0
DSNB777I -   ASYNCHRONOUS WRITES
               CHANGED PAGES                          = 0
               CLEAN PAGES                            = 0
             FAILED WRITES DUE TO LACK OF STORAGE     = 0
```

```
DSNB778I - CASTOUT THRESHOLDS DETECTED
             FOR CLASSES                              = 0
             FOR GROUP BUFFER POOL                    = 0
           GBP CHECKPOINTS TRIGGERED                  = 0
           PARTICIPATION IN REBUILD                   = 1
DSNB796I - CASTOUTS
             PAGES CASTOUT                            = 0
             UNLOCK CASTOUT                           = 0
             READ CASTOUT CLASS                       = 0
             READ CASTOUT STATISTICS                  = 0
             READ DIRECTORY INFO                      = 0
DSNB797I - OTHER INTERACTIONS
             REGISTER PAGE                            = 0
             UNREGISTER PAGE                          = 0
             DELETE NAME                              = 0
             READ STORAGE STATISTICS                  = 0
             EXPLICIT CROSS INVALIDATIONS             = 0
             ASYNCHRONOUS GBP REQUESTS               = 0
DSNB764I - DUPLEXING STATISTICS FOR GBP29-SEC
             WRITES

               FAILED DUE TO LACK OF STORAGE          = 0
               ASYNCHRONOUS COMPLETION CHECKS         = 0
DSNB793I -   DELETE NAME LIST                         = 0
             READ CASTOUT STATISTICS                  = 0
             DELETE NAME                              = 0
             OTHER ASYNCHRONOUS GBP REQUESTS          = 0
DSNB790I - DISPLAY FOR GROUP BUFFER POOL GBP29 IS COMPLETE
DSN9022I - DSNB1CMD '-DISPLAY GBPOOL' NORMAL COMPLETION
```

Chapter 27. -DISPLAY LOCATION (DB2)

The DISPLAY LOCATION command displays various information about the specified remote locations. If you specify the DETAIL option, each line can be followed by information regarding conversations owned by DB2 system threads that are communicating with the location.

The information returned by the DISPLAY LOCATION command reflects a dynamic status. By the time the information is displayed, it is possible that the status has changed.

Abbreviation: -DIS LOC

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 178
- "Option descriptions" on page 178
- "Output" on page 178
- "Examples" on page 179

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

Authorization

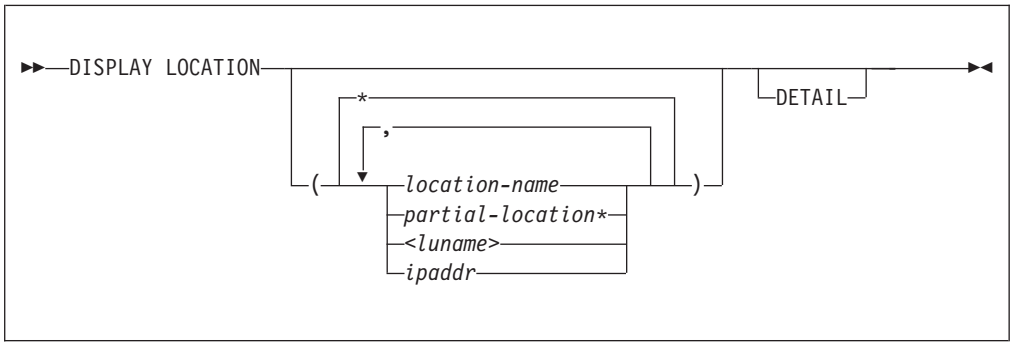
To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- DISPLAY privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

(*) Displays information for all remote locations.

(*location-name***)**
Lists one or more location names, separated by commas.

Because DB2 does not receive a location name from requesters that are not DB2 UDB for z/OS subsystems, you can enter the LUNAME or IP address of such a requester. Refer to the option descriptions for the **<***luname***>** and **(***ipaddr***)** options for more information about using the LUNAME or IP address to specify a requester that is not a BD2 for z/OS subsystem.

(*partial-location****)**
Selects all location names that begin with the string *partial-location* and can end with any string, including the empty string. For example, LOCATION(ABC*) selects all location names that begin with the string 'ABC'.

<*luname***>**
Requests information about the remote clients that are connected to DDF through the remote SNA LU that is specified. Enclose the LU name in the less-than (<) and greater-than (>) symbols. For example, DISPLAY LOCATION(<LULA>) displays information about a remote location (that is not DB2 UDB for z/OS) with the LU name of LULA.

You can use an asterisk (*) when specifying an LU name in the same manner as previously described for specifying a partial-location name. For example, DISPLAY LOCATION(<LULA*) selects all remote locations (that are not DB2 UDB for z/OS) with an LU name that begins with the string 'LULA'.

(*ipaddr***)**
Requests information about the clients that are connected to DDF through the remote TCP/IP host. Enter the IP address in the form *nnn.nnn.nnn.nnn*. For example, DISPLAY LOCATION(124.63.51.17) displays information about clients at the remote TCP/IP host whose dotted decimal IP address is 124.63.51.17.

DETAIL
Displays additional information about conversation activity for DB2 system threads, as shown in “Example 2” on page 179.

Output

The DISPLAY LOCATION command displays the following output:

LOCATION The LOCATION of the remote system.

PRDID The product identifier (PRDID) of the remote system. The PRDID is displayed in the form *nnnvvrrm*, where:

nnn The database product
vv The product version
rr The product release
m The product modification level

LINKNAME The address (LU name or IP address) of the remote system.

REQUESTERS
The number of active threads from the local subsystem that are accessing the remote system.

SERVERS The number of threads from the remote system that are accessing the local subsystem.

CONVERSATIONS
The total number of conversations or sockets related to the partner system.

Examples

Example 1: Display information about threads and conversations with specific remote locations, using the following command:

```
-DISPLAY LOCATION(SAN_JOSE,SAN_FRANCISCO)

DSNL200I - DISPLAY LOCATION REPORT FOLLOWS-
LOCATION       PRDID       LINKNAME           REQUESTERS SERVERS CONVS
SAN_JOSE       DSN05010    LUND1                   1         0      1
SAN_FRANCISCO  DSN05010    LUND3                   1         0      1
DISPLAY LOCATION REPORT COMPLETE
```

Example 2: Display information about threads and conversations with all remote locations. Additionally, display detail conversation information about DB2 system threads that communicate with other locations. This is an example of the output generated by the following command:

```
-DISPLAY LOCATION DETAIL

DSNL200I - DISPLAY LOCATION REPORT FOLLOWS-
LOCATION          PRDID     LINKNAME           REQUESTERS SERVERS CONVS
SAN_JOSE          DSN05010  LUND1                   1         0      3
 -SYSTASK   SESSID           A ST TIME
 -SYSCON-O  00D359691359EE80   S  9128009214880
 -SYSCON-I  00D359691359EE81 W R  9128009214881
MENLO_PARK        DSN05010  LUND2                   1         0      4
 -SYSTASK   SESSID           A ST TIME
 -SYSCON-O  00D359691359EE82   S  9128009214882
 -SYSCON-I  00D359691359EE83 W R  9128009214883
 -RESYNC    00D359691359EE84 V R  9128009214884
SAN_FRANCISCO     DSN05010  LUND3                   1         0      6
 -SYSTASK   SESSID           A ST TIME
 -SYSCON-O  0000000000000000   C  9128009214885
 -SYSCON-I  00D359691359EE86 W R  9128009214886
 -RESYNC    00D359691359EE87 W R  9128009214887
 -RESYNC    00D359691359EE88 W R  9128009214888
 -RESYNC    00D359691359EE89 W R  9128009214889
DISPLAY LOCATION REPORT COMPLETE
```

Example 3: Display information for a DB2 system that is connected to the following DRDA partners:

- A non-z/OS server named DRDALOC via TCP/IP.
- Several TCP/IP clients from the same TCP/IP host as the DRDALOC server.
- A DB2 for z/OS server named DB2SERV via SNA.

```
DISPLAY LOCATION(*)

DSNL200I - DISPLAY LOCATION REPORT FOLLOWS -
 LOCATION      PRDID     LINKNAME        REQUESTERS  SERVERS   CONVS
 DRDALOC       SQL03030  124.63.51.17    3           0         3
 124.63.51.17  SQL03030  124.63.51.17    0           15        15
 DB2SERV       DSN05010  LULA            1           0         1
 DISPLAY LOCATION REPORT COMPLETE
```

Example 4: The following example assumes DB2 is connected to the following DRDA partners:

- DB2A is connected to this DB2 system, using TCP/IP for DRDA connections and SNA for DB2 private protocol connections.
- DB2SERV is connected to this DB2 using only SNA.

```
DISPLAY LOCATION(*)

DSNL200I - DISPLAY LOCATION REPORT FOLLOWS -
 LOCATION      PRDID     LINKNAME        REQUESTERS  SERVERS   CONVS
 DB2A          DSN05010  LUDB2A          3           4         9
 DB2A          DSN05010  124.38.54.16    2           1         3
 DB2SERV       DSN04010  LULA            1           1         3
 DISPLAY LOCATION REPORT COMPLETE
```

Chapter 28. -DISPLAY LOG (DB2)

The DB2 command DISPLAY LOG displays log information about, and the status of, the offload task.

Abbreviation: DIS LOG

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Usage notes"
- "Examples" on page 182

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

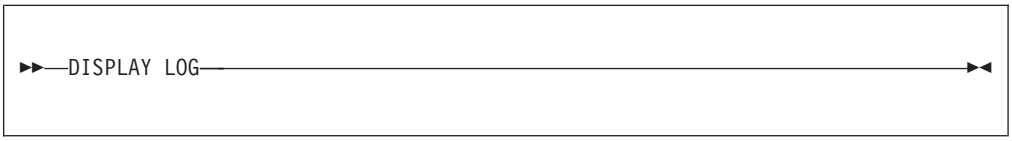
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- DISPLAY privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Usage notes

Information provided by the DISPLAY LOG command: You can use the DISPLAY LOG command to view the current LOGLOAD setting, including information about the current active log data sets and status of the offload task. You can obtain additional information about log data sets and checkpoint information by using the Print Log Map utility (DSNJU004). For more information about the Print Log Map utility, see Part 3 of *DB2 Utility Guide and Reference*.

© Copyright IBM Corp. 1983, 2008

Examples

Example 1: Display log information and status of the offload task.

```
-DISPLAY LOG
```

This command produces output similar to the following output:

```
DSNJ370I - DSNJCOOA LOG DISPLAY
CURRENT COPY1 LOG = DSNC810.LOGCOPY1.DS03 IS 22% FULL
CURRENT COPY2 LOG = DSNC810.LOGCOPY2.DS03 IS 22% FULL
H/W RBA = 0000039A9F24, LOGLOAD = 150000
FULL LOGS TO OFFLOAD = 2 OF 6, OFFLOAD TASK IS (BUSY,ALLC)
DSNJ371I - DB2 RESTARTED 14:06:23 MAY 22, 2002
RESTART RBA 0000039A8000
DSN9002I - DSNJC001 'DIS LOG' NORMAL COMPLETION
```

This example shows the following information:

- The active log data sets are 22% full. If you are running dual logs and the percentages are different, the log data sets are of different sizes. DB2 switches both active logs when one reaches the end of the file. This can result in unused active log space if one log data set is larger than the other.
- The current LOGLOAD setting is 150000 log records between system checkpoints. You can modify this value using the SET LOG command.
- Two of the six active log data sets require archiving. The status of the offload task includes the indicator that it is busy, allocating an archive log data set. This might be an indication of an outstanding tape mount on the system console. If the status remains busy and no longer seems to be functioning, you can terminate the task and then restart it using the ARCHIVE LOG CANCEL OFFLOAD command.
- DB2 was started at 14:06:23 on MAY 22, 2002, and began logging at RBA 0000039A8000.

Chapter 29. -DISPLAY PROCEDURE (DB2)

The DB2 command DISPLAY PROCEDURE displays statistics about stored procedures that are accessed by DB2 applications. This command displays one line of output for each stored procedure that a DB2 application has accessed. You can qualify stored procedure names with a schema name.

The information returned by the DISPLAY PROCEDURE command reflects a dynamic status. By the time the information is displayed, it is possible that the status could have changed.

Abbreviation: -DIS PROC

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 184
- "Option descriptions" on page 184
- "Output" on page 185
- "Examples" on page 186

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or a CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group or local, depending on the value of the SCOPE option.

Authorization

To run this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- Ownership of the stored procedure
- DISPLAY privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

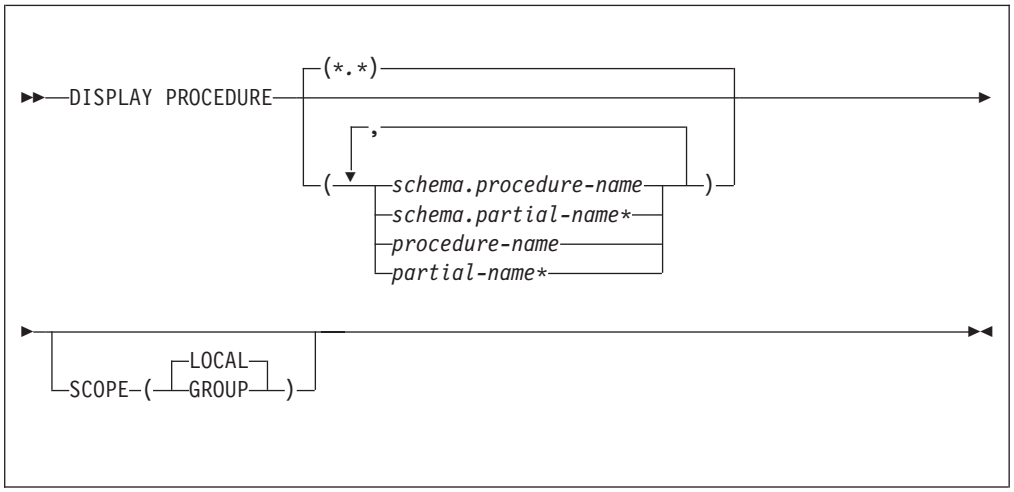
If you specify DISPLAY PROCEDURE *.* or *schema.partial-name**, the privilege set of the process must include one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

(*.*)
Displays information for all stored procedures in all schemas that DB2 applications have accessed since DB2 was started.

(*schema.procedure-name***)**
Displays the specified stored procedure in the specified schema.

(*schema.partial-name****)**
Displays a set of stored procedures in the specified schema that DB2 applications have accessed since DB2 was started. The names of all procedures in the set begin with *partial-name* and can end with any string, including the empty string. For example, PAYROLL.ABC* displays information for all stored procedure names beginning with ABC in the PAYROLL schema.

(*procedure-name***)**
Displays one or more specific stored procedure names in the SYSPROC schema. If no procedures are named, DB2 displays information for all stored procedures that have been accessed by DB2 applications.

(*partial-name****)**
Displays information for a set of stored procedures in the SYSPROC schema that DB2 applications have accessed since DB2 was started. The names of all procedures in the set begin with *partial-name* and can end with any string, including the empty string. For example, ABC* displays information for all stored procedures in the SYSPROC schema with names that begin with ABC.

SCOPE
Specifies the scope of the command.

(LOCAL)
Specify to display information about procedures on the local member only.

(GROUP)
Specify to display information about procedures on all members of the data sharing group.

Output

Sample output: The DISPLAY PROCEDURE command generates output similar to the following output:

```
DSNX940I = DSNX9DIS DISPLAY PROCEDURE REPORT FOLLOWS -

PROCEDURE      STATUS    ACTIVE  QUED MAXQ TIME FAIL WLM_ENV
APPL1          STARTED     1       0    0    0    1  SANDBOX
APPL2          STARTED     1       0    0    0    0  SANDBOX
APPL2          STARTED     0       1    2    0    0  SANDBOX
APPL5          STOPREJ     0       0    0    0    0  SANDBOX
APPL6          STOPABN     0       0    0    0    0  SANDBOX
PROC1          STOPQUE     0       0    0    0    0  SANDBOX

DSNX9DIS DISPLAY PROCEDURE REPORT COMPLETE
DSN9022I = DSNX9COM '-DISPLAY PROC' NORMAL COMPLETION
```

Description of output: Each output line displays the following information:

PROCEDURE
The name of the stored procedure.

STATUS
The status of the stored procedure of the stored procedure. The possible values are:

STARTED	Requests for the procedure can be processed.
STOPQUE	Requests are queued.
STOPREJ	Requests are rejected.
STOPABN	Requests are rejected because of abnormal termination.

ACTIVE
The number of threads that are currently running the load module.

QUED
The number of threads that are waiting for the procedure to be scheduled.

MAXQ
The maximum number of threads that have waited concurrently for the procedure to be scheduled since DB2 was started. DB2 resets this value to 0 each time you run the START PROCEDURE command.

TIME The number of times an SQL CALL statement timed out while waiting for a request for the procedure to be scheduled. DB2 resets this value to 0 each time you run the START PROCEDURE command.

FAIL The number of times a procedure has failed. DB2 resets this value to 0 each time you run the START PROCEDURE command.

Message DSNX943I lists a range of procedures that are stopped because a STOP PROCEDURE command included a partial name with a pattern-matching character (*), as in the following example:

```
-STOP PROCEDURE(ABC*)
```

Message DSNX950I is returned when DISPLAY PROCEDURE is issued for a procedure name that has not been accessed by a DB2 application.

Examples

Example 1: Display information about all stored procedures that have been accessed by DB2 applications.

```
-DISPLAY PROCEDURE
```

This command produces output similar to the following output:

```
DSNX940I = DSNX9DIS DISPLAY PROCEDURE REPORT FOLLOWS-
PROCEDURE      STATUS    ACTIVE    QUED  MAXQ  TIMEOUT  FAIL   WLM_ENV
USERPRC1       STARTED     0         0     1      0       0    SANDBOX
USERPRC2       STOPQUE     0         5     5      3       0    SANDBOX
USERPRC3       STARTED     2         0     6      0       0    SANDBOX
USERPRC4       STOPREJ     0         0     1      0       0    SANDBOX
DSNX9DIS DISPLAY PROCEDURE REPORT COMPLETE
DSN9022I = DSNX9COM '-DISPLAY PROC' NORMAL COMPLETION
```

Example 2: Display information about specific stored procedures in the SYSPROC schema.

```
-DISPLAY PROCEDURE(SYSPROC.USERPRC2,USERPRC4)
```

This command produces output similar to the following output:

```
DSNX940I = DSNX9DIS DISPLAY PROCEDURE REPORT FOLLOWS-

------ SCHEMA=SYSPROC
PROCEDURE      STATUS    ACTIVE    QUED  MAXQ  TIMEOUT  FAIL   WLM_ENV
USERPRC2       STOPQUE     0         5     5      3       0     SANDBOX
USERPRC4       STOPREJ     0         0     1      0       0     SANDBOX
DSNX9DIS DISPLAY PROCEDURE REPORT COMPLETE
DSN9022I = DSNX9COM '-DISPLAY PROC' NORMAL COMPLETION
```

Example 3: Display information about stored procedures in the PAYROLL and HRPROD schemas.

```
-DISPLAY PROCEDURE(PAYROLL.*,HRPROD.*)
```

This command produces output similar to the following output:

```
DSNX940I = DSNX9DIS DISPLAY PROCEDURE REPORT FOLLOWS-

------ SCHEMA=PAYROLL
PROCEDURE      STATUS    ACTIVE    QUED  MAXQ  TIMEOUT  FAIL   WLM_ENV
PAYPRC1        STARTED     0         0     1      0       0     PAYROLL
PAYPRC2        STOPQUE     0         5     5      3       0     PAYROLL
PAYPRC3        STARTED     2         0     6      0       0     PAYROLL
USERPRC4       STOPREJ     0         0     1      0       0     SANDBOX

------ SCHEMA=HRPROD
PROCEDURE      STATUS    ACTIVE    QUED  MAXQ  TIMEOUT  FAIL   WLM_ENV
HRPRC1         STARTED     0         0     1      0       0     HRPROCS
HRPRC2         STOPREJ     0         0     1      0       0     HRPROCS
DSNX9DIS DISPLAY PROCEDURE REPORT COMPLETE
DSN9022I = DSNX9COM '-DISPLAY PROC' NORMAL COMPLETION
```

Example 4: Display information about all stored procedures in the SYSADM schema that have been accessed by DB2 applications. Assume that the -STOP PROCEDURE(SYSADM.SP*) ACTION(QUEUE) command is in effect at the time that the following command is issued.

```
-DISPLAY PROCEDURE(SYSADM.*)
```

This command produces output similar to the following output:

```
DSNX940I = DSNX9DIS DISPLAY PROCEDURE REPORT FOLLOWS-

------ SCHEMA=SYSADM
PROCEDURE      STATUS    ACTIVE    QUED  MAXQ  TIMEOUT  FAIL   WLM_ENV
SPC1           STOPQUE     0         0    0      0       0     WLMENV1
SPC2           STOPQUE     0         0    0      0       0     WLMENV3
DSNX9DIS PROCEDURES SP - SP* STOP QUEUE
DSNX9DIS DISPLAY PROCEDURE REPORT COMPLETE
DSN9022I = DSNX9COM '-DISPLAY PROC' NORMAL COMPLETION
```

Chapter 30. -DISPLAY RLIMIT (DB2)

The DB2 command DISPLAY RLIMIT displays the current status of the resource limit facility (governor). If the facility has already been started, DISPLAY RLIMIT also displays the ID of the resource limit specification table that is being used.

Abbreviation: -DIS RLIM

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Example"

Environment

This command can be issued from a z/OS console, a DSN session, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax

Example

Display the current status of the resource limit facility.

```
-DISPLAY RLIMIT
```

If the resource limit facility (RLF) is inactive, the following output is generated:

```
DSNT701I - RESOURCE LIMIT FACILITY IS INACTIVE
DSN9022I - DSNTCDIS 'DISPLAY RLIMIT' NORMAL COMPLETION
```

© Copyright IBM Corp. 1983, 2008

If the RLF is active, the value of field RESOURCE AUTHID on panel DSNTIPP is SYSADM, and the resource limit specification table with
RLST NAME SUFFIX = 03 was started, the following output is generated:

```
DSNT700I = SYSADM.DSNRLST03 IS THE ACTIVE RESOURCE LIMIT
SPECIFICATION TABLE
DSN9022I = DSNTCDIS 'DISPLAY RLIMIT' NORMAL COMPLETION
```

Chapter 31. -DISPLAY THREAD (DB2)

The DB2 command DISPLAY THREAD displays current status information about DB2 threads. A DB2 thread can be an allied thread, a database access thread, or a parallel task thread. Threads can be active, inactive, indoubt, or postponed.

Distributed threads are those threads that have a connection with a remote location (active or inactive) or that had a connection with a remote location (indoubt). An allied thread and a parallel task thread can be distributed or non-distributed; a database access thread is always distributed.

The DISPLAY THREAD command allows you to select the type of information you want to display by using one or more of the following criteria:

- Active threads, inactive threads, indoubt threads, postponed threads, or the set of active, indoubt, and postponed threads (see the descriptions under the TYPE option for more information)
- Allied threads, including those threads that are associated with the address spaces whose connection names are specified
- Distributed threads, including those threads that are associated with a specific remote location
- Detailed information about connections with remote locations
- A specific logical unit of work ID (LUWID)

The information that is returned by the DISPLAY THREAD command reflects a dynamic status. When the information is displayed, it is possible that the status has changed. Moreover, the information is consistent only within one address space and is **not necessarily** consistent across all address spaces displayed.

Abbreviation: -DIS THD

The following topics provide additional information:

- "Environment"
- "Authorization" on page 192
- "Syntax" on page 192
- "Option descriptions" on page 193
- "Usage notes" on page 196
- "Output" on page 197
- "Examples" on page 199

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group or local, depending on the SCOPE option.

© Copyright IBM Corp. 1983, 2008

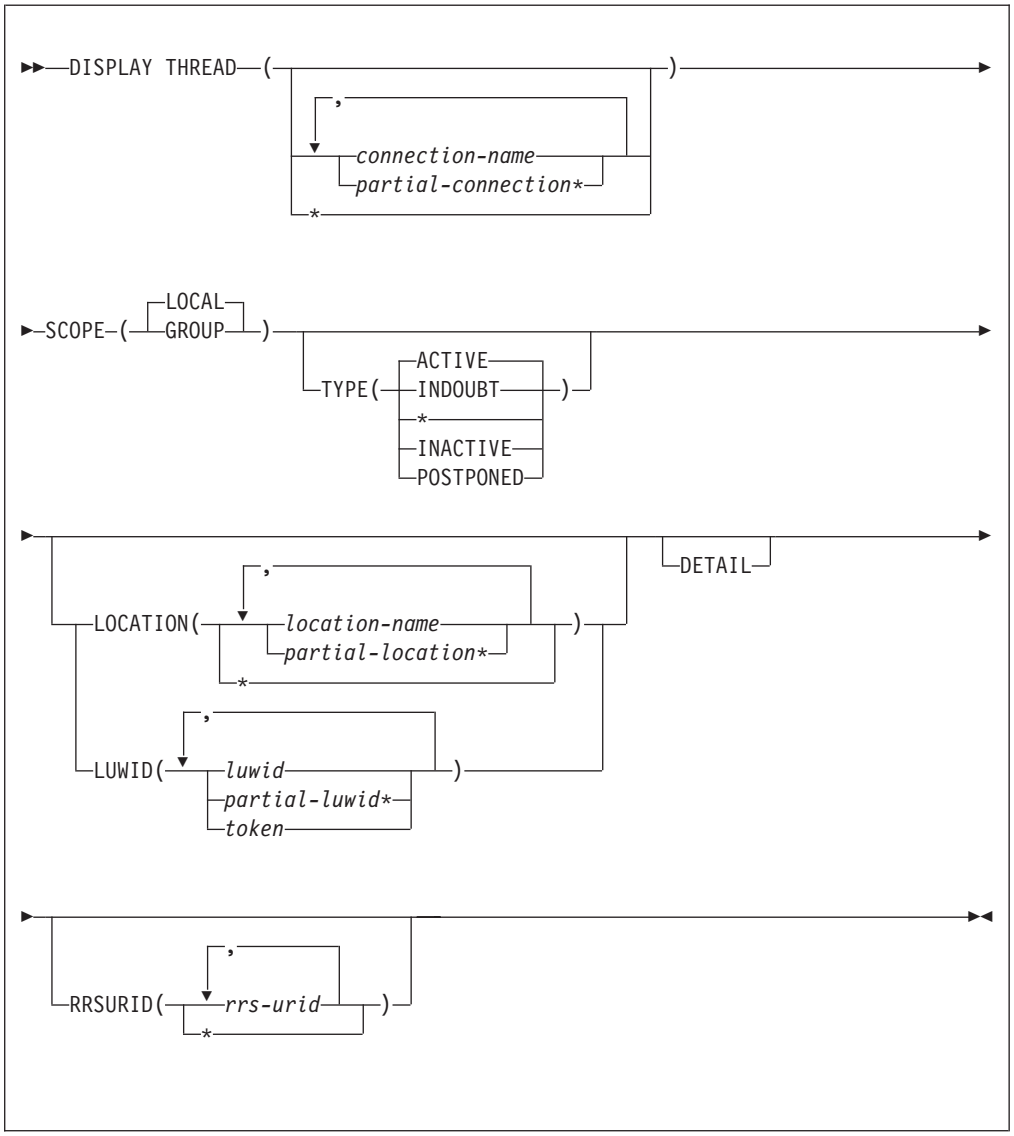
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- DISPLAY privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Option descriptions

Only under certain conditions, as described in the following lists, are any of the following options required.

If you do not specify either (*connection-name*) or (*), the following rules apply:

- If the command is issued from a DSN session under TSO, a DB2I panel (DB2 COMMANDS), or an IMS or CICS terminal, the connection name is inherited from the associated address space.
- If the command is not issued from one of those environments, the following rules apply:
  - If you do not specify either LOCATION or LUWID, processing terminates with a DSNV413I message.
  - If you do specify LOCATION or LUWID, only distributed threads of the type selected by the TYPE option are displayed.
  - When you explicitly specify *location-name*, only distributed threads of the type selected by the TYPE option that either have (active or inactive threads) or had (indoubt threads) a connection with the specified location are displayed.

(*connection-name***, ...)**
Lists one or more connection names (of 1 to 8 characters each). Allied threads are selected only from the address spaces associated with those connection names. The LOCATION option can restrict what is displayed:

- If you specify LOCATION(*), only distributed threads of the type specified in the TYPE option are displayed.
- When you explicitly specify *location-name*, only distributed threads of the specified type that either have or had a connection with the specified location are displayed.

(*partial-connection****, ...)**
Selects the connections that begin with the string *partial-connection* and can end with any string, including the empty string. For example, DISPLAY THREAD(CICS*,IMS*) selects all connection names that begin with the string 'CICS' or 'IMS'. The LOCATION option can restrict the display exactly the same way as previously described for *location-name*.

(*)
Displays all threads in all address spaces attached to DB2 and all database access threads of the types specified in the TYPE option. The LOCATION option can restrict what is displayed:

- If you specify LOCATION(*), only distributed threads are displayed.
- When you explicitly specify *location-name*, only distributed threads that either have (active or inactive threads) or had (indoubt threads) a connection with the specified location are displayed.

The **default** is to display only the connections that are associated with the transaction manager from which the command was entered.

SCOPE
Specifies the scope of the command.

(LOCAL)
Displays threads on only the current member.

(GROUP)
Displays all threads on the data sharing group.

TYPE
Tells the type of thread to display.

Abbreviation: T

(ACTIVE)
Displays only active threads. An active allied thread is connected to DB2 via TSO, BATCH, IMS, CICS or CAF. An active database access thread is connected via VTAM to another system and is performing work on behalf of that system. If, during command processing, an active thread becomes indoubt, it can appear twice—once as active and once as indoubt.

Abbreviation: A

The information that is produced by ACTIVE can be useful for debugging purposes, especially messages DSNV403I and DSNV404I; the contents of those messages are described in Part 2 of *DB2 Messages*.

(INDOUBT)
Displays only indoubt threads.

An indoubt thread is a participant in a two-phase commit protocol that has completed the first phase of commit, and has then lost communication with the commit coordinator and does not know whether to commit or roll back the updates that have been made.

The indoubt thread information that is displayed includes threads for which DB2 has a coordinator role, a participant role, or both coordinator and participant roles.

The commit coordinator for an allied thread is either a transaction manager (for example, IMS or CICS) or z/OS RRS for threads that use RRSAF. The commit coordinator for a database access thread is a requester at a remote system.

Indoubt threads hold locks on all resources that were updated.

Abbreviation: I

(*) Displays active, indoubt, and postponed threads.

(INACTIVE)
Displays only inactive threads. An inactive thread is a database access thread that is connected via VTAM to another system and is idle, waiting for a new unit of work to begin from that system.

Abbreviation: INA

Use qualifiers such as complete location names or LUWIDs with this option. When there are large numbers of inactive database access threads, unqualified display requests can temporarily change the DB2 working set, which can temporarily affect the performance of active threads.

(POSTPONED)
Displays information about units of work whose back-out processing has been postponed.

Abbreviation: P

After you have identified postponed threads, use the RECOVER POSTPONED command, described on page 317, to complete backout processing for the postponed units of work.

LOCATION(*location-name***, ...)**
Limits the display to distributed threads as described.

Abbreviation: LOC

location-name
Displays only distributed threads of the specified type that either have (active or inactive threads) or had (indoubt threads) a remote connection with the specified *location-name*.

DB2 does not receive a location name from requesters that are not DB2 UDB for z/OS subsystems. To display information about a requester that is not a DB2 UDB for z/OS subsystem, enter its LU name or IP address. Enclose the LU name in the less-than (<) and greater-than (>) symbols. Enter the IP address in the form *nnn.nnn.nnn.nnn*. For example, the following command displays information about a remote location (that is not DB2 UDB for z/OS) with the LU name of LULA:

```
-DISPLAY THREAD (*) LOCATION (<LULA>)
```

The following command displays information about a remote location (that is not DB2 UDB for z/OS) with an IP address of 123.34.101.98:

```
-DISPLAY THREAD (*) LOCATION (123.34.101.98)
```

DB2 uses the <LU name> notation or IP address in messages displaying information about requesters other than DB2.

*partial-location**
Selects all location names that begin with the string *partial-location* and can end with any string, including the empty string. For example, `LOCATION(SAN*)` selects all location names that begin with the string 'SAN'.

You can use an asterisk (*) when specifying an LU name in the same manner as previously described for other location names that are not DB2 UDB for z/OS subsystems. For example, `LOCATION(<LULA*)` selects all remote locations (that are not DB2 UDB for z/OS) with an LU name that begins with the string 'LULA'.

You cannot use an asterisk when you specify an IP address.

(*) Display all distributed threads of the specified type.

LUWID(*luwid***, ...)**
Displays information about the distributed threads that have the specified LUWID. It is possible for more than one thread to have the same LUWID.

luwid
Consists of a fully qualified LU network name followed by a period and an LUW instance number.

The LU network name consists of a one- to eight-character network ID, a period, and a one- to eight-character network LU name. The LUW instance number consists of 12 hexadecimal characters that uniquely identify the unit of work.

*partial-luwid**
Selects all LUWIDs that begin with the string *partial-luwid* and can end with any string, including the empty string. For example, LUWID(NET1.*) selects all LUWIDs with a network name of 'NET1'.

token
Identifies a specific thread in an alternate way. DB2 assigns a token to each distributed thread it creates. A token is a one- to six-digit decimal number that appears after the equal sign in all DB2 messages that display a LUWID.

If you do not include any periods nor a '*' in the LUWID specification, DB2 assumes that you are supplying a token. The token that DB2 assigns to a specific LUWID is unique for that DB2 subsystem, but not necessarily unique across subsystems.

DETAIL
Displays additional information about active, inactive, and indoubt threads.

RRSURID(*rrs-urid***)**
Specifies that only threads that match the specified RRSURID selection criteria are to be displayed.

- If RRSURID(*rrs-urid*) is specified, any thread involved in the RRSURID that has the value *rrs-urid*, and that meets any other specified selection criteria, will be displayed.
- If RRSURID(*) is specified, any thread involved in any RRSURID, and that meets any other specified selection criteria, will be displayed.

Usage notes

Formatted report for distributed threads: The series of messages, DSNV444I through DSNV446I, augment the formatted report for DISPLAY THREAD TYPE (ACTIVE or INACTIVE) for distributed threads. Refer to these messages in Part 2 of *DB2 Messages* for an explanation of the formatted report.

Threads using private protocol and DRDA access: A database access thread that is connected to a requester can also be connected to another database server location using DB2 private protocol access or DRDA access. In this case, DB2 issues message DSNV445I for the requester, and message DSNV444I and zero or more DSNV446I messages for the remote server connections. In this case, the database access thread acts as an intermediate database server.

Participant threads waiting for the commit or abort decision: A DSNV465I message is issued for an active participant thread that has completed phase 1 of commit processing and has been waiting for the commit or abort decision from the coordinator for more than 60 seconds.

DISPLAY THREAD output limit: If a DISPLAY THREAD command is issued from the z/OS console, the maximum number of lines of output for a single invocation of the command is 255 lines (at which time a DSNV421I or DSNV422I message is printed). If you do not receive the required information in the first 255 lines of output, issue the command again, specifying the TYPE option and a specific connection name, location, luwid, or a combination of these, as appropriate, to reduce the output.

Showing parallel tasks: The DISPLAY THREAD command shows parallel tasks by using a status type of PT. The parallel tasks are displayed immediately after the originating task. If the thread has a status of PT, the connection name contains blanks if the thread of the originating task is running on the same DB2 subsystem. This shows that these parallel tasks are related to the originating task. If the parallel task is running on a DB2 subsystem that is different from the subsystem that runs the originating task, the connection name is shown and the entry is followed by message DSNV443I.

Displaying the XID: If the DISPLAY THREAD command is issued with the TYPE ACTIVE and DETAIL options, or with the TYPE INDOUBT option, message DSNV440I displays the contents of the XID. The contents of the XID are displayed as a hexadecimal value.

The XID is displayed in the DISPLAY THREAD TYPE INDOUBT report if the indoubt transaction is XID related.

Output

Table 19 shows sample DISPLAY THREAD commands and the types of output they generate. Numbers in each row refer to the descriptions at the bottom of the table. The DETAIL keyword is not included because it affects only the amount of information displayed about a distributed thread.

If RRSURID(*) is specified, only threads involved in any RRSURID will be displayed. If RRSURID(*rrs-urid*) is specified, only threads involved in that specific RRSURID will be displayed.

Table 19. Sample DISPLAY THREAD commands. The following output is generated when commands are issued from different environments with different TYPE specifications. (Specifying TYPE(*) displays the equivalent output of both TYPE(ACTIVE) and TYPE(INDOUBT) in one report.)

Environment where the command is issued	Command	ACTIVE	INDOUBT	INACTIVE
A DSN session under TSO, DB2I, IMS, or CICS, where the connection name is inherited	-DIS THD	1	1	2
	-DIS THD LOC(*)	3	3	2
	-DIS THD LOC(*location-name*)	4	4	2
z/OS console	-DIS THD	6	6	6
	-DIS THD LOC(*)	9	9	8
	-DIS THD LOC(*location-name*)	10	10	11
Any source	-DIS THD(*connection-name*)	1,12	1,12,15	12
	-DIS THD(*connection-name*) LOC(*)	3,12	3,12,15	12
	-DIS THD(*connection-name*) LOC(*location-name*)	4,13	4,13,15	13
	-DIS THD(*)	7	7,15,16	8
	-DIS THD(*) LOC(*)	9	9,15,16	8
	-DIS THD(*) LOC(*location-name*)	10	10,15,16	11
	-DIS THD(*) LUWID(*luwid* or *token*)	5	5,15,16	5
	-DIS THD(*connection-name*) LUWID(*luwid* or *token*)	14	14,15,16	14
	-DIS THD LUWID(*luwid* or *token*)	5	5,15,16	5

Table 19. Sample DISPLAY THREAD commands (continued). The following output is generated when commands are issued from different environments with different TYPE specifications. (Specifying TYPE(*) displays the equivalent output of both TYPE(ACTIVE) and TYPE(INDOUBT) in one report.)

Environment where the command is issued	Command	ACTIVE	INDOUBT	INACTIVE
Description of display that is generated: 1. Allied threads of the specified TYPE with the connection name. 2. No threads (inactive threads are database access threads and have no inherited connection name). 3. Distributed allied threads of the specified TYPE with the connection name. 4. Distributed allied threads of the specified TYPE with the connection name and a distributed connection = *location-name*. 5. The threads of the specified TYPE that have LUWID = *luwid* or *token*. 6. Message DSNV413I is displayed to indicate an error. 7. All threads (both allied and database access) of the specified TYPE. 8. All inactive database access threads. 9. All distributed threads (both allied and database access) of the specified TYPE. 10. All distributed threads (both allied and database access threads) of the specified TYPE with a distributed connection = *location-name*. 11. All inactive database access threads with a distributed connection = *location-name*. 12. Database access threads of the specified TYPE with the connection name. 13. Database access threads of the specified TYPE with the connection name and a distributed connection = *location-name*. 14. A thread of the specified TYPE with the connection name and LUWID = *luwid* or *token*. 15. Messages DSNV407 and DSNV408 also display coordinator's TCP/IP resync port number; message DSNV446 also displays the participant's TCP/IP resync port number. 16. Message DSNV440I displays the contents (in hexadecimal representation) of the remote client's XID for distributed threads of the specified TYPE that are associated with an XA transaction manager coordinator.				

If the DETAIL option is specified, the following additional information is displayed:

LOCATION
: The location name of the remote system.

SESSID
: For a VTAM connection, the VTAM defined session instance identifier of the session on which the conversation is executing.

  For a TCP/IP connection, the local and remote TCP/IP port numbers, in the form *local:remote*. *local* is the port number for the local DB2 subsystem. *remote* is the port number for the remote partner.

A
: If VTAM or TCP/IP has control of the conversation (if DB2 transferred control of the thread to VTAM or TCP/IP for that conversation), a V is displayed in the A (Active) column. W indicates that the DB2 subsystem has suspended processing on this conversation until VTAM notifies the DB2 subsystem that the VTAM event is complete. The column is otherwise blank.

STATUS
: This 2-byte column indicates the status of the conversation or socket. The possible values for STATUS are:

  Value Status

S*x* Send
R*x* Receive
A*x* Allocation
D*x* Deallocation
C*x* Change number of sessions (CNOS) processing
X*x* Exchange Log name processing
blank Not in one of the preceding states

x can be one of the following values:
1 = Private protocol conversation with single-phase commit
2 = DRDA conversation with single-phase commit
3 = Private protocol conversation with two-phase commit
4 = DRDA conversation with two-phase commit.

If you specify the DETAIL keyword for active, inactive, or indoubt threads, DB2 also displays message DSNV441I, which displays accounting suffix information as a character string. DB2 will recognize accounting information, and display suffix information that originates from the following systems:

- DB2 UDB for z/OS systems
- Any system where the accounting information matches the format that is described in DSNSQMDA.QMDASQLI, which includes the product ID prefixes JCC (DB2 Universal JDBC driver) and SQL (DB2 Universal Database for Linux, UNIX, and Windows).

If you specify the DETAIL option for active threads, DB2 also displays message DSNV440I, which displays the contents (in hexadecimal representation) of the remote client's XID.

Examples

Example 1: The output of the command DISPLAY THREAD shows a token for every thread, distributed or not. This example shows the token for an allied thread that is not distributed. The token is 123. You can use the thread's token as the parameter in the CANCEL THREAD command.

```
-DISPLAY THREAD(*) DETAIL
```

This command produces output similar to the following output:

```
DSNV401I - DISPLAY THREAD REPORT FOLLOWS -
DSNV402I - ACTIVE THREADS -
NAME     ST A   REQ ID               AUTHID    PLAN      ASID TOKEN
BATCH    T  *     5 BKH2C            SYSADM    BKH2      000D   123
DISPLAY ACTIVE REPORT COMPLETE
DSN9022I - DSNVDT '-DIS THD' NORMAL COMPLETION
```

Example 2: This example shows information about conversation activity when distribution information is displayed for active threads. DB2 returns the following output, indicating that the local site application is waiting for a conversation to be allocated in DB2, and that a DB2 server is accessed by a DRDA client using TCP/IP.

```
-DISPLAY THREAD(*) LOCATION(*) DETAIL
```

This command produces output similar to the following output:

```
DSNV401I - DISPLAY THREAD REPORT FOLLOWS -
DSNV402I - ACTIVE THREADS -
     NAME     ST A   REQ ID           AUTHID   PLAN     ASID   TOKEN
     TSO      TR *     3 SYSADM       SYSADM   DSNESPRR 002E       2
      V436-PGM=DSNESPRR.DSNESM68, SEC=1, STMNT=116
```

```
      V444-DB2NET.LUND0.A238216C2FAE=2 ACCESSING DATA AT
      V446-USIBMSTODB22:LUND1
      V447--LOCATION        SESSID        A ST    TIME
      V448--USIBMSTODB22    0000000000000000 V A1 9015816504776
     TSO      RA *    11 SYSADM       SYSADM   DSNESPRR 001A      15
      V445-STLDRIV.SSLU.A23555366A29=15 ACCESSING DATA FOR 123.34.101.98
      V447--LOCATION        SESSID           A ST    TIME
      V448--123.34.101.98   446:3171        S2    9015611253108
DISPLAY ACTIVE REPORT COMPLETE
DSN9022I - DSNVDT '-DIS THD' NORMAL COMPLETION
```

Example 3: In this example, a system at Site 1 has a TSO application and an IMS application. The system at Site 1 fails after DB2 commits the TSO application, but before the commit decision has been communicated to the participant subsystems at Site 2 and Site 3. The failure occurs before IMS has communicated the commit or rollback decision to the Site 1 DB2 subsystem. The DISPLAY THREAD commands are issued after the Site 1 DB2 subsystem restarts but before reconnect with IMS. DISPLAY THREAD commands that are issued at each location show output similar to the following output:

The following DISPLAY THREAD command is issued at Site 1:

```
-DISPLAY THREAD(*) TYPE(INDOUBT)
```

This command produces output similar to the following output:

```
DSNV401I - DISPLAY THREAD REPORT FOLLOWS -
DSNV406I - INDOUBT THREADS -
 COORDINATOR               STATUS     RESET URID         AUTHID
 STLIMS01                  INDOUBT          0F201050A010 SM09H
  V467-HAS LUWID IBM.STLDB21.15A86A876789.0010=1
  V449-HAS NID=A5 AND ID=STLIMS01
  V450-HAS PARTICIPANT INDOUBT AT
  V446--IBMSJ0DB20001:STLDB22
 IBMSTLDB20001             COMMITTED        0F20105B0000 JO78S
  V467-HAS LUWID IBM.STLDB21.16B57B954427.0003=2
  V450-HAS PARTICIPANT INDOUBT AT
  V446--IBMSJ0DB20001:STLDB22  IBMLA0DB20001:STLDB23
 DISPLAY INDOUBT REPORT COMPLETE -
 DSN9022I - DSNVDT '-DIS THD' NORMAL COMPLETION
```

The following DISPLAY THREAD command is issued at Site 2:

```
-DISPLAY THREAD(*) TYPE(INDOUBT)
```

This command produces output similar to the following output:

```
 DSNV401I - DISPLAY THREAD REPORT FOLLOWS -
 DSNV406I - INDOUBT THREADS -
 COORDINATOR               STATUS     RESET URID         AUTHID
 IBMSTLDB20001:STLDB21     INDOUBT          03201050A010 HEU4443
  V467-HAS LUWID IBM.STLDB21.15A86A876789.0010=8
  V466-THREAD HAS BEEN INDOUBT FOR 00:05:20
 IBMSTLDB20001:STDB21      INDOUBT          03201050B000 PP433MM
  V467-HAS LUWID IBM.STLDB21.16B57B954427.0003=6
 DISPLAY INDOUBT REPORT COMPLETE
 DSN9022I - DSNVDT '-DIS THD' NORMAL COMPLETION
```

The following DISPLAY THREAD command is issued at Site 3 (where both ACTIVE and INDOUBT threads are displayed):

```
-DISPLAY THREAD(*) TYPE(*) DETAIL
```

This command produces output similar to the following output:

```
DSNV401I - DISPLAY THREAD REPORT FOLLOWS -
DSNV402I - ACTIVE THREADS -
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
SERVER   RA *     0 RUW2STAT     JONES    DISTSERV 0005     4
 V465-THREAD HAS BEEN PREPARED FOR 00:05:20
 V445-IBM.STLDB21.15A86A876789=4 ACCESSING DATA FOR
      IBMSJ0DB20001:STLDB21
 V447--LOCATION         SESSID           A ST TIME
 V448--IBMSJ0DB20001    0000000400000004 W R4 9034817015032
DISPLAY ACTIVE REPORT COMPLETE
DSNV406I - INDOUBT THREADS -
COORDINATOR               STATUS     RESET URID         AUTHID
IBMSTLDB20001:STLDB21     INDOUBT          03201050B000 SM43YY33
 V467-HAS LUWID IBM.STLDB21.16B57B954427.0003=5
 V466-THREAD HAS BEEN INDOUBT FOR 00:05:20
DISPLAY INDOUBT REPORT COMPLETE
DSN9022I - DSNVDT '-DIS THD' NORMAL COMPLETION
```

Example 4: This example shows a thread executing within a stored procedure and a thread waiting for a stored procedure to be scheduled. Assume that an application makes a call to stored procedure PROC1 and then to stored procedure PROC2. PROC2 is in a STOP QUEUE state.

The output for PROC1 while it is executing shows a status of SP in the ST column, which indicates that a thread is executing within a stored procedure:

```
-DISPLAY THREAD(*)
```

This command produces output similar to the following output:

```
DSNV401I - DISPLAY THREAD REPORT FOLLOWS -
DSNV402I - ACTIVE THREADS - 176
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
BATCH    SP       3 RUNAPPL      SYSADM   PL01AP01 001D    43
 V429 CALLING STORED PROCEDURE PROC1, LOAD MODULE LMPROC1
DISPLAY ACTIVE REPORT COMPLETE
DSN9022I - DSNVDT '-DISPLAY THREAD' NORMAL COMPLETION
```

The output for PROC2, while it is queued, shows a status of SW in the ST column, which indicates that a thread is waiting for a stored procedure to be scheduled:

```
-DISPLAY THREAD(*)
```

This command produces output similar to the following output:

```
DSNV401I - DISPLAY THREAD REPORT FOLLOWS -
DSNV402I - ACTIVE THREADS - 198
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
BATCH    SW *    13 RUNAPPL      SYSADM   PL01AP01 001D    43
 V429 CALLING STORED PROCEDURE PROC2, LOAD MODULE
DISPLAY ACTIVE REPORT COMPLETE
DSN9022I - DSNVDT '-DISPLAY THREAD' NORMAL COMPLETION
```

Example 5: This example shows an allied, nondistributed originating thread (TOKEN=30) that is established (allocated according to plan) in addition to all of its parallel tasks (PT), which are running on the same DB2 system. All parallel tasks are displayed immediately following their corresponding originating thread.

```
16.32.57       DB1G DISPLAY THREAD(*)
16.32.57 STC00090  DSNV401I DB1G DISPLAY THREAD REPORT FOLLOWS -
16.32.57 STC00090  DSNV402I DB1G ACTIVE THREADS -
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
BATCH    T  *     1 PUPPYDML     USER001  DSNTEP3  0025    30
         PT *   641 PUPPYDML     USER001  DSNTEP3  002A    40
         PT *    72 PUPPYDML     USER001  DSNTEP3  002A    39
         PT *   549 PUPPYDML     USER001  DSNTEP3  002A    38
```

```
         PT *   892 PUPPYDML     USER001  DSNTEP3  002A    37
         PT *    47 PUPPYDML     USER001  DSNTEP3  002A    36
         PT *   612 PUPPYDML     USER001  DSNTEP3  002A    35
         PT *   545 PUPPYDML     USER001  DSNTEP3  002A    34
         PT *   432 PUPPYDML     USER001  DSNTEP3  002A    33
         PT *   443 PUPPYDML     USER001  DSNTEP3  002A    32
         PT *   252 PUPPYDML     USER001  DSNTEP3  002A    31
DISPLAY ACTIVE REPORT COMPLETE
16.32.58 STC00090  DSN9022I DB1G DSNVDT '-DISPLAY THREAD' NORMAL
 COMPLETION
```

Example 6: This example shows the detail report for a DB2 client that uses TCP/IP to access a remote DRDA server.

```
-DISPLAY THREAD(*) LOCATION(*)
```

This command produces output similar to the following output:

```
DSNV401I - DISPLAY THREAD REPORT FOLLOWS -
DSNV402I - ACTIVE THREADS -
   NAME     ST A   REQ ID           AUTHID PLAN     ASID   TOKEN
   BATCH    TR *     6 BKH2C        SYSADM YW1019C  0009      2
    V444-STLDRIV.SSLU.A23555366A29=2 ACCESSING DATA AT
    V446-USIBMSTODB22:123.34.101.98:446
    V447--LOCATION         SESSID           A ST TIME
    V448--USIBMSTODB22     4019:446         V R2 9015611253116
   DISPLAY ACTIVE REPORT COMPLETE
DSN9022I - DSNVDT '-DIS THD' NORMAL COMPLETION
```

Example 7: This example shows the detail report for a DB2 server that is accessed by a DRDA client using TCP/IP.

```
DISPLAY THREAD(*) LOCATION(*)
```

This command produces output similar to the following output:

```
DSNV401I - DISPLAY THREAD REPORT FOLLOWS -
DSNV402I - ACTIVE THREADS -
   NAME     ST A   REQ ID           AUTHID PLAN     ASID   TOKEN
   BATCH    RA *     5 BKH2C        SYSADM DISTSERV 0008      2
    V445-STLDRIV.SSLU.A23555366A29=2 ACCESSING DATA FOR 123.34.101.98
    V447--LOCATION         SESSID           A ST TIME
    V448--123.34.101.98    446:3171           S2 9015611253108
   DISPLAY ACTIVE REPORT COMPLETE
DSN9022I - DSNVDT '-DIS THD' NORMAL COMPLETION
```

Example 8: This example shows information about units of work whose back-out processing has been postponed.

```
-DISPLAY THREAD (*) TYPE (POSTPONED)
```

This command produces output similar to the following output:

```
DSNV401I - DISPLAY THREAD REPORT FOLLOWS -
DSNV431I - POSTPONED ABORT THREADS -
COORDINATOR              STATUS      RESET URID         AUTHID
BATCH                    ABORT-P           000002FF98EA ADMF001
BATCH                    ABORT-P           000002FF9000 ADMF001
DISPLAY POSTPONED ABORT REPORT COMPLETE
DSN9022I - DSNVDT '-DISPLAY THREAD' NORMAL COMPLETION
```

Example 9: This example shows the token for a thread that is executing a user-defined function. The token is 18.

```
-DISPLAY THREAD(*) DETAIL
```

This command produces output similar to the following output:

```
DSNV401I - DISPLAY THREAD REPORT FOLLOWS -
DSNV402I - ACTIVE THREADS -
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
BATCH    T  *   231 DISTHD       ADMF001           0021   95
BATCH    SW *    38 INSERT       ADMF001  DSNTEP3  0025   18
 V436-PGM=CLIP74C1.UFIP74C1, SEC=0, STMNT=0
 V429 CALLING FUNCTION =SCIP7401.SP_UFIP74C1       ,
     PROC=V61AWLM3, ASID=0030, WLM_ENV=WLMENV3
DISPLAY ACTIVE REPORT COMPLETE
DSN9022I - DSNVDT '-DISPLAY THREAD' NORMAL COMPLETION
```

Example 10: This example shows information about a thread that is involved in an RRS unit of recovery.

```
-DISPLAY THREAD(*) RRSURID(*)
```

This command produces output similar to the following output:

```
- 08.23.58 STC00149  DSNV401I  ( DISPLAY THREAD REPORT FOLLOWS -
   - 08.23.58 STC00149  DSNV402I  ( ACTIVE THREADS -
   - NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
   - RRSAF    T        8 TGXID-111    ADMF001  TGXIDR   0023    35
   -  V481-DB2 IS PARTICIPANT FOR RRS URID B4D0FC267EB020000000001101010000
   - DISPLAY ACTIVE REPORT COMPLETE
   - 08.23.58 STC00149  DSN9022I  ( DSNVDT '-DIS THD' NORMAL COMPLETION
```

Example 11: This example shows information about a thread where DB2 is the coordinator for an indoubt RRS unit of recovery. DB2 has committed the thread but has not been able to resolve the RRS UR with RRS.

```
-DISPLAY THREAD(*) TYPE(I) RRSURID(*)
```

This command produces output similar to the following output:

```
- 09.27.21 STC00185  DSNV406I  ( INDOUBT THREADS -
   - COORDINATOR               STATUS      RESET URID         AUTHID
   - UNKNOWN                   COMMITTED         123456789ABC UNKNOWN
   -  V480-DB2 IS COORDINATOR FOR RRS URID C4D4FA267EB040000000001201020000
 00- DISPLAY INDOUBT REPORT COMPLETE
   - 09.27.21 STC00185  DSNV434I  ( DSNVDT NO POSTPONED ABORT THREADS FOUND
   - 09.27.21 STC00185  DSN9022I  ( DSNVDT '-DIS THD' NORMAL COMPLETION
```

Example 12: This example shows the XID for an active thread that is associated with an XA transaction manager:

```
-DISPLAY THREAD(*) DETAIL
```

This command produces output similar the following output:

```
#dis thd(*) det
DSNV401I  # DISPLAY THREAD REPORT FOLLOWS -
DSNV402I  # ACTIVE THREADS -
NAME     ST A   REQ ID           AUTHID    PLAN       ASID TOKEN
TEST0001 RX *     2 CTHDCORID001 SYSADM    DONSQL1    0036    12
 V440-XID=53514C20 00000017 00000000 544D4442
          00000000 002F93DD A92F8C4F F3000000
          0000BD
 V445-USIBMSY.SYEC715A.B5A0992212F4=12 ACCESSING DATA FOR
      9.30.115.130:5001
 V447--LOCATION          SESSID           A ST TIME
 V448--STL715A           E15FE002DB8DEDCD W R4 0109210383564
DISPLAY ACTIVE REPORT COMPLETE
DSN9022I  # DSNVDT '-DIS THD' NORMAL COMPLETION
```

Example 13: This example shows information about an active database access
thread that is disconnected.

```
# -DISPLAY THREAD(*) DETAIL
```

This command produces output similar to the following output:

```
13.23.46          =dis thread(*) detail
13.23.46 STC00067  DSNV401I  = DISPLAY THREAD REPORT FOLLOWS -
13.23.46 STC00067  DSNV402I  = ACTIVE THREADS -
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
DISCONN  DA *     8 NONE         NONE     DISTSERV 003B    11
 V471-USIBMSY.SYEC1DB2.C1C38BDE809D=11
DISPLAY ACTIVE REPORT COMPLETE
13.23.46 STC00067  DSN9022I  = DSNVDT '-DIS THREAD' NORMAL COMPLETION
```

Example 14: This example shows information about an inactive connection.

```
-DISPLAY THREAD(*) TYPE(INACTIVE)
```

This command produces output similar to the following output:

```
- 13.27.47          =dis thread(*) type(inactive)
- 13.27.47 STC00067  DSNV401I  = DISPLAY THREAD REPORT FOLLOWS -
- 13.27.47 STC00067  DSNV424I  = INACTIVE THREADS -
- NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
- SERVER   R2       0 javaw.exe    ADMF001  DISTSERV 003B    12
-  V437-WORKSTATION=KFUKUSH, USERID=admf001,
-       APPLICATION NAME=javaw.exe
-  V445-G91E1686.EF04.080107212323=12 ACCESSING DATA FOR 9.30.22.134
- SERVER   R2       0 db2bp.exe    ADMF001  DISTSERV 003B    14
-  V437-WORKSTATION=KFUKUSH, USERID=admf001,
-       APPLICATION NAME=db2bp.exe
-  V445-G91E1686.F104.080107212343=14 ACCESSING DATA FOR 9.30.22.134
- DISPLAY INACTIVE REPORT COMPLETE
- 13.27.47 STC00067  DSN9022I  = DSNVDT '-DIS THREAD' NORMAL COMPLETION
```

Chapter 32. -DISPLAY TRACE (DB2)

The DB2 command DISPLAY TRACE displays a list of active traces. For more information about this trace facility, see Part 4 (Volume 1) of *DB2 Administration Guide*.

An additional option to this command and additional values for a few options of this command are not described here. They are intended for service and use under the direction of IBM support personnel. For details, see *DB2 Diagnosis Guide and Reference*.

Abbreviation: -DIS TRACE

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 206
- "Option descriptions" on page 207
- "Examples" on page 209

Environment

This command can be issued from a z/OS console, a DSN session, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group or local, depending on the value of the SCOPE option.

Authorization

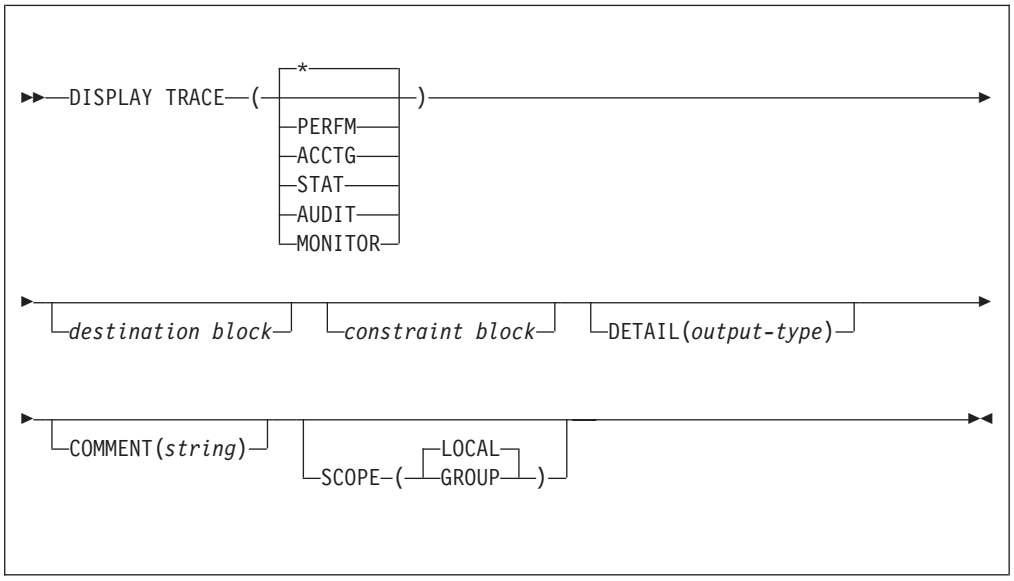
To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- DISPLAY privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

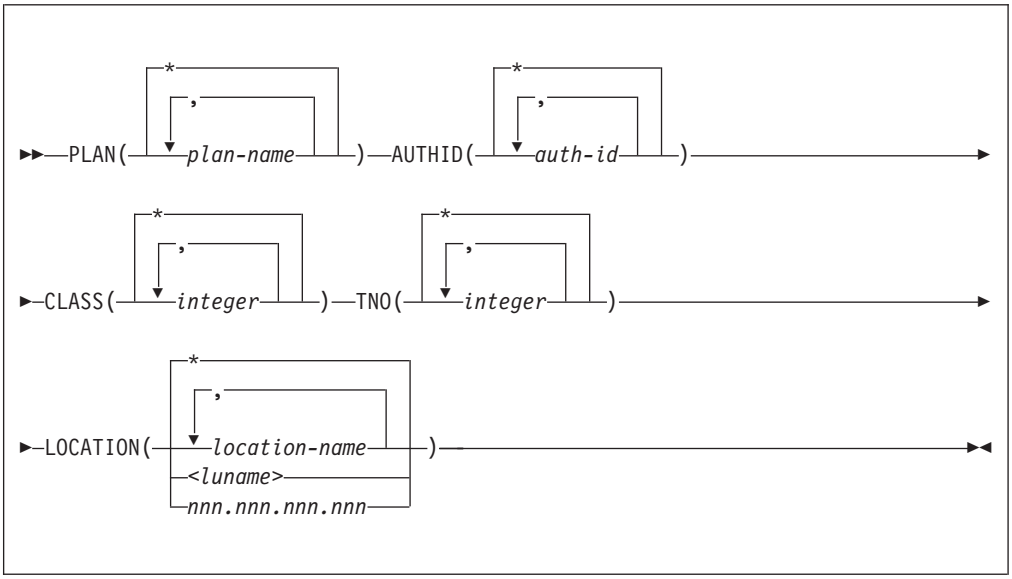
© Copyright IBM Corp. 1983, 2008

Syntax



destination block:

constraint block:



Option descriptions

None of the options are required. The command DISPLAY TRACE lists all active traces. Each option that is used, except TNO, limits the effect of the command to active traces that were started using the same option, either explicitly or by default, with exactly the same parameter values. For example, the following command lists only the active traces that were started using the options PERFM *and* CLASS (1,2); it does *not* list, for example, any trace started using CLASS(1).

```
-DISPLAY TRACE (PERFM) CLASS (1,2)
```

(*) Does not limit the list of traces. The **default** is (*).

The CLASS option cannot be used with DISPLAY TRACE (*).

Each of the following keywords limits the list to traces of the corresponding type. For further descriptions of each type, see Chapter 77, "-START TRACE (DB2)," on page 383.

Type (Abbrev)	Description
PERFM (P)	Performance records of specific events
ACCTG (A)	Accounting records for each transaction
STAT (S)	Statistical data
AUDIT (AU)	Audit data
MONITOR (MON)	Monitor data

DETAIL(*output-type***)**
Limits the information that a trace displays based on the output type specified within parentheses.

The possible values for *output-type* are:

1 Display summary trace information: TRACE NUMBER, TYPE, CLASS, DEST

2 Display qualification trace information: TRACE NUMBER, AUTHID, PLAN, LOCATION

1,2 Display both summary and qualification information

* Display both summary and qualification information

If no parameter follows DETAIL, type 1 trace information is displayed.

An additional column, QUAL, is also displayed, indicating whether the trace is qualified. Part of the summary trace information, the QUAL column can be used to determine if further qualification information for the trace is available. This information can be obtained by specifying DETAIL (2) or DETAIL (*). A QUAL column value of YES indicates that additional information for this particular trace exists in the qualification trace information; a value of NO indicates that no additional information for this trace exists.

COMMENT(*string***)**
Specifies that comment *string* appears in the trace output, except for the output in the resident trace tables.

string is any character string; it must be enclosed between apostrophes if it includes a blank, comma, or special character. The comment does not appear in the display; it can be recorded in trace output, but only if commands are being traced.

SCOPE
Specifies the scope of the command.

(LOCAL)
Displays the trace for the local member only.

(GROUP)
Displays the trace for all members in the data sharing group.

DEST
Limits the list to traces started for particular destinations. More than one value can be specified, but do not use the same value twice. If you do not specify a value for DEST, DB2 does not use the destination of where trace output is recorded to limit the list of traces displayed.

Abbreviation: D

Possible values and their meanings are:

Value	Trace destination
GTF	The generalized trace facility
SMF	The system management facility
SRV	An exit to a user-written routine
OP*n*	A specific destination. *n* can be a value from 1 to 8.

See Chapter 77, "-START TRACE (DB2)," on page 383 for a list of allowable destinations for each trace type.

PLAN(*plan-name***, ...)**
Limits the list to traces started for particular application plans. Up to eight plan names can be used. If more than one name is used, only one value can be used for AUTHID, TNO, and LOCATION. Do not use this option with STAT.

The **default** is **PLAN**(*), which does not limit the list.

AUTHID(*authorization-id***, ...)**
Limits the list to traces started for particular authorization identifiers. Up to

eight identifiers can be used. If more than one identifier is used, only one value can be used for PLAN, TNO, and LOCATION. Do not use this option with STAT.

The **default** is **AUTHID(*)**, which does not limit the list.

CLASS(*integer***, ...)**
Limits the list to traces started for particular classes. For descriptions of the allowable classes, see Chapter 77, "-START TRACE (DB2)," on page 383.

The **default** is **CLASS(*)**, which does not limit the list.

TNO(*integer***, ...)**
Limits the list to particular traces, identified by their trace numbers (1 to 32, 01 to 09). Up to eight trace numbers can be used. If more than one number is used, only one value each for PLAN, AUTHID, and LOCATION can be used.

The **default** is **TNO(*)**, which does not limit the list.

LOCATION(*location-name***, ...)**
Limits the list to traces started for threads that have a distributed relationship with the specified location.

(*location-name***)**
The location names that you supply are the 1- to 16-character identifiers assigned to the DB2 subsystem whose traces you want to display. Supplying an * as the location name indicates that the trace display must include all traces started with any location name qualifier.

You can specify up to eight location names. If you specify more than one location name, you can only specify one value each for PLAN, AUTHID, and TNO.

LOCATION cannot be specified when you choose a statistics trace.

Requesters other than DB2 UDB for z/OS: DB2 does not receive a location name from requesters that are not DB2 UDB for z/OS subsystems. To display information about a requester that is not a DB2 UDB for z/OS subsystem, enter its LU name, enclosed in the less-than (<) and greater-than (>) symbols. For example, the following command displays information about a remote location with the LU name of LULA:

```
-DISPLAY TRACE (*) LOCATION (<LULA>)
```

DB2 uses the < LU name> notation in messages displaying information about requesters that are not DB2 UDB for z/OS subsystems.

The **default** is **LOCATION(*)**, which does not limit the list.

<luname>
Activates the DB2 trace for the remote clients that are connected to DDF through the remote SNA LU that you specified in *luname*.

nnn.nnn.nnn.nnn
Activates the DB2 trace for the remote clients that are connected to DDF through the remote TCP/IP host whose IP address is specified by *nnn.nnn.nnn.nnn*.

Examples

Example 1: List all traces that have the generalized trace facility as their only destination.

```
-DISPLAY TRACE (*) DEST (GTF)
```

Example 2: List the trace started for Example 2 of the START TRACE command, which is shown on page 210.

```
-DISPLAY TRACE (ACCTG) PLAN (DSN8BC81)
  COMMENT ('ACCTG TRACE FOR DSN8BC81')
```

Example 3: List all active performance traces.

```
-DISPLAY TRACE=P
```

Example 4: List all active audit traces for threads that are connected to the DB2 subsystem with location name USIBMSTODB23.

```
-DISPLAY TRACE (AUDIT) LOCATION (USIBMSTODB23)
```

Example 5: Output from the DISPLAY TRACE command is a set of messages that look similar to the following messages:

```
- 10.26.34           -DISPLAY TRACE
- 10.26.34 STC   21  DSNW127I - CURRENT TRACE ACTIVITY IS -
- TNO TYPE   CLASS        DEST QUAL
- 01  STAT   01           SMF  NO
- 02  ACCTG  01           SMF  YES
- 03  PERFM  01,02,03     GTF  YES
- 04  AUDIT  01,02,03,04, SMF  YES
- 04         06,07
- 05  MON    01,02,03     OP1  NO
- *********END OF DISPLAY TRACE SUMMARY DATA*********
- 10.26.34 STC   21  DSN9022I - DSNWVCM1 '-DISPLAY TRACE' NORMAL COMPLETION

- 10.28.47           -DISPLAY TRACE DETAIL(*)
- 10.28.47 STC   21  DSNW127I - CURRENT TRACE ACTIVITY IS -
- TNO TYPE   CLASS        DEST QUAL
- 01  STAT   01           SMF  NO
- 02  ACCTG  01           SMF  YES
- 03  PERFM  01,02,03     GTF  YES
- 04  AUDIT  01,02,03,04, SMF  YES
- 04         06,07
- 05  MON    01,02,03     OP1  NO
- *********END OF DISPLAY TRACE SUMMARY DATA*********
- 10.28.47 STC   21  DSNW143I - CURRENT TRACE QUALIFICATIONS ARE -
- TNO AUTHID   PLAN     RMID         LOCATION
- 01  *        *        *
- 02  *        *        *
- 03  USER01   *        *
- 04  *        *        14,16,18,26  DENVER
- 05  *        PROG1    *
- 06  *        *        *
- ******END OF DISPLAY TRACE QUALIFICATION DATA******
- 10.28.47 STC   21  DSN9022I - DSNWVCM1 '-DISPLAY TRACE' NORMAL COMPLETION
```

Chapter 33. -DISPLAY UTILITY (DB2)

The DB2 command DISPLAY UTILITY displays the status of utility jobs, including utility jobs in a data sharing group.

The output from the command consists of informational messages only. One set of messages is returned for each job identified by the command. For utility jobs in a data sharing group, the output shows the member name of the system on which each utility job is running.

The status from the display represents the current status, except in a data sharing group when the utility is running on a member other than the one from which the command is issued. In that case, the status is current as of the last checkpoint.

Abbreviation: -DIS UTIL

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions" on page 212
- "Usage notes" on page 212
- "Output" on page 212
- "Examples" on page 214

Environment

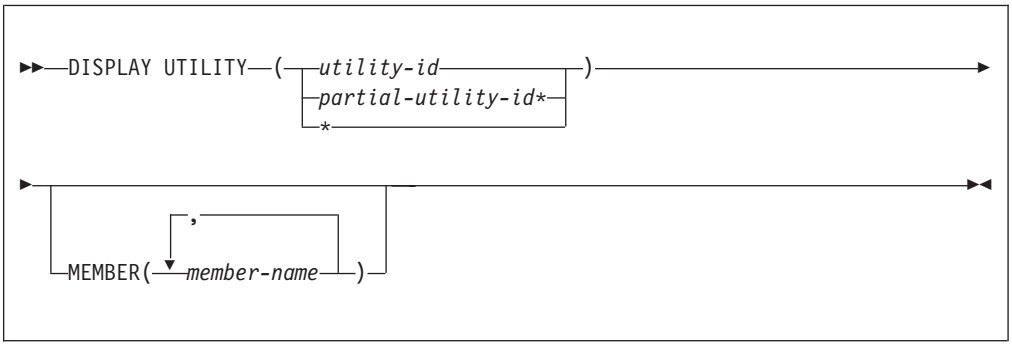
This command can be issued from a z/OS console, a DSN session, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group or member, depending on which option you choose

Authorization

None is required.

Syntax



© Copyright IBM Corp. 1983, 2008

Option descriptions

Use at least one of the following options but do not use the same option more than once.

(*utility-id***)**
Identifies a single job by its utility identifier, the value given for the UID parameter when the job was created.

If *utility-id* was created by the DSNU CLIST by default, it has the form of *tso-userid.control-file-name*. For a list of values for *control-file-name*, see the description of the UID parameter for the DSNU command procedure (CLIST) in *DB2 Utility Guide and Reference*.

If *utility-id* was omitted when the utility job was created, *utility-id* has the form *userid.jobname*.

(*partial-utility-id****)**
Identifies a set of utility jobs. A status message is shown for each utility identifier that begins with the characters of *partial-utility-id*.

For example, `-DISPLAY UTILITY(ABCD*)` shows the status of every utility job known to DB2 whose identifier begins with the characters ABCD.

(*) Shows the status of all utility jobs known to DB2, including jobs currently running in a data sharing group.

MEMBER (*member-name***, ...)**
Restricts the display for the identified utility jobs to specific members of the data sharing group. The default is to display utility jobs running on any member. In a non-data-sharing environment, the option is ignored.

One set of messages is returned for each job identified by the command.

Usage notes

DISPLAY status: The status displayed in the returned message is the status at the time the DB2 utility function received the command. Execution has proceeded, therefore the current state of the utility can be different from the state reported. For instance, the DISPLAY UTILITY command can indicate that a particular utility identifier is active, but, when the message is received by the requester, the utility job step could have terminated so that the utility identifier is no longer known to DB2.

Command response: In a data sharing environment, messages DSNU100I, DSNU105I, DSNU106I show the name of the member on which the utility job is running. If you specify a single member name in the MEMBER option and that member does not belong to the group, or if you specify a list of member names in the MEMBER option and none of those members belong to the group, the command fails and a message is issued.

Output

The output from the DISPLAY UTILITY command consists of informational messages only.

Output during any phase of REORG with SHRLEVEL CHANGE or SHRLEVEL REFERENCE: During *any* phase of REORG with SHRLEVEL CHANGE or SHRLEVEL REFERENCE, the output of DISPLAY UTILITY includes the

information in DSNU347I. During *any* phase of REORG with SHRLEVEL CHANGE, the output of DISPLAY UTILITY includes information in DSNU384I as shown in "Example 4" on page 214.

DEADLINE
: Indicates a timestamp according to the most recently specified value of DEADLINE.

MAXRO
: Indicates the number of seconds, according to the most recently specified value of MAXRO.

LONGLOG
: Indicates either CONTINUE, TERM, or DRAIN according to the most recently specified value of LONGLOG.

DELAY
: Indicates the number of seconds according to the most recently specified value of DELAY.

Output during LOG phase of REORG with SHRLEVEL CHANGE: During the LOG phase of REORG with SHRLEVEL CHANGE, the output of DISPLAY UTILITY now includes the additional information found in message DSNU383I, as shown in "Example 4" on page 214.

CURRENT ITERATION NUMBER
: Indicates the current iteration number.

WRITE ACCESS ALLOWED IN CURRENT ITERATION
: Indicates "YES" or "NO" according to whether write access is allowed in the current iteration of log processing.

ITERATION BEFORE PREVIOUS ITERATION
: Indicates the ELAPSED TIME so far, and the NUMBER OF LOG RECORDS PROCESSED in the iteration. Their value is zero if the current iteration number is one or two.

PREVIOUS ITERATION
: Indicates the ELAPSED TIME and the NUMBER OF LOG RECORDS PROCESSED for the previous iteration. Their value is zero if the current iteration number is one.

CURRENT ITERATION:
: Indicates the ESTIMATED ELAPSED TIME, the ACTUAL ELAPSED TIME SO FAR and the ACTUAL NUMBER OF LOG RECORDS BEING PROCESSED.

CURRENT ESTIMATE FOR NEXT ITERATION
: For the next iteration, indicates the currently ELAPSED TIME and the currently estimated NUMBER OF LOG RECORDS TO BE PROCESSED.

Progress of utility processing: The DISPLAY UTILITY command provides the user with an estimate of how much processing the utility has completed. The output displays information from message DSNU105I as seen in "Example 2" on page 214, and includes the following information:

COUNT
: COUNT *n* is the number of pages or records processed in a utility phase. COUNT has different meanings for different utilities. For utilities not mentioned in the following list, ignore this field.

- For the LOAD utility, COUNT represents the total number of records that have been loaded into all partitions when the command is issued. The count is zero from the time the RELOAD phase starts until the first LOAD subtask begins loading records into the first partition assigned to that subtask.
- For the CHECK INDEX, RECOVER INDEX, and REORG utilities, COUNT represents the number of records processed. For the CHECK INDEX, RECOVER INDEX, and REORG utilities, COUNT represents the number of pages processed. If the DISPLAY UTILITY command is entered from the other member, then where REORG is running, the COUNT number remains zero in the REORG BUILD or SORTBLD phase.
- For the COPY, MERGE COPY, RECOVER (restore phase), and RUNSTATS utilities, COUNT represents the number of pages processed.
- For the STOSPACE utility, COUNT represents the number of table spaces or indexes processed.

For more information, refer to *DB2 Utility Guide and Reference*.

Examples

Example 1: Display status information for all utility jobs currently known to DB2.

```
-DISPLAY UTILITY (*)
```

Example 2: Display the status of utilities jobs on all members of the data sharing group.

```
-DB1G DISPLAY UTILITY (*)
```

The following output, which shows utility jobs on members DB1G and DB2G, is generated:

```
 DSNU100I -DB1G DSNUGDIS USER = SAMPID
                         MEMBER = DB1G
                         UTILID = RUNTS
                         PROCESSING UTILITY STATEMENT 1
                         UTILITY = RUNSTATS
                         PHASE = RUNSTATS  COUNT = 0
                         STATUS = STOPPED
 DSNU100I -DB1G DSNUGDIS USER = SAMPID
                         MEMBER = DB2G
                         UTILID = CHKIX1
                         PROCESSING UTILITY STATEMENT 8
                         UTILITY = CHECK
                         PHASE = UNLOAD  COUNT = 0
                         STATUS = STOPPED
 DSN9022I -DB1G DSNUGCC  '-DB1G DISPLAY UTILITY' NORMAL COMPLETION
```

Example 3: In a data sharing environment, display the status of utilities on member DB1G.

```
-DB1G DISPLAY UTILITY (*) MEMBER (DB1G)
```

Example 4: This shows output from the following DISPLAY UTILITY command:

```
-DB1G DISPLAY UTILITY(*)
DSNU105I -DB1G DSNUGDIS - USERID = SYSADM 973
                MEMBER = DB1G
                UTILID = REORGCP
                PROCESSING UTILITY STATEMENT 1
                UTILITY = REORG
                PHASE = LOG   COUNT = 0
```

```
                STATUS = ACTIVE
DSNU347I -DB1G DSNUGDIS - 974
                DEADLINE = NONE
DSNU384I -DB1G DSNUGDIS - 975
                MAXRO = DEFER
                LONGLOG = CONTINUE
                DELAY = 1200 SECONDS
DSNU383I -DB1G DSNUGDIS - CURRENT ITERATION NUMBER = 4 976
WRITE ACCESS ALLOWED IN THIS ITERATION = YES
ITERATION BEFORE PREVIOUS ITERATION:
   ELAPSED TIME = 00:00:00
   NUMBER OF LOG RECORDS PROCESSED = 0
PREVIOUS ITERATION:
   ELAPSED TIME = 00:00:00
   NUMBER OF LOG RECORDS PROCESSED = 0
CURRENT ITERATION:
   ESTIMATED ELAPSED TIME = 00:00:00
   ACTUAL ELAPSED TIME SO FAR = 00:00:00
   ACTUAL NUMBER OF LOG RECORDS BEING PROCESSED = 0
CURRENT ESTIMATE FOR NEXT ITERATION:
   ELAPSED TIME = 00:00:00
   NUMBER OF LOG RECORDS TO BE PROCESSED = 0
SN9022I -DB1G DSNUGCCC '-DIS UTIL' NORMAL COMPLETION
```

Chapter 34. DSN (TSO)

The TSO command DSN enables you to issue the following DSN subcommands:
- ABEND
- BIND
- DCLGEN
- END
- FREE
- REBIND
- RUN
- SPUFI

During a DSN session, you can enter DB2 commands or comments. DB2 commands must start with a hyphen (-). Comments must start with an asterisk (*).

During a DSN session, you can also issue TSO commands, except for FREE, RUN, TEST, and TIME. To use TSO TEST to debug an application program, run it with the DSN command; for example:

```
TEST 'prefix.SDSNLOAD(DSN)' CP
```

The ABEND subcommand is used for diagnostic purposes only, and is intended to be used only under the direction of IBM Software Support. Use it only when diagnosing a problem with DSN or DB2. Percent commands are not recognized during a DSN session, they are only supported by the TSO command processor.

The following topics provide additional information:
- "Environment"
- "Authorization"
- "Syntax" on page 218
- "Option descriptions" on page 218
- "Usage notes" on page 218
- "Examples" on page 219

Environment

A DSN session runs under TSO in either foreground or background mode. When you run it in background mode, you are not prompted for corrections or additional required information.

You can also start a DSN session from a CLIST running in either foreground or background mode.

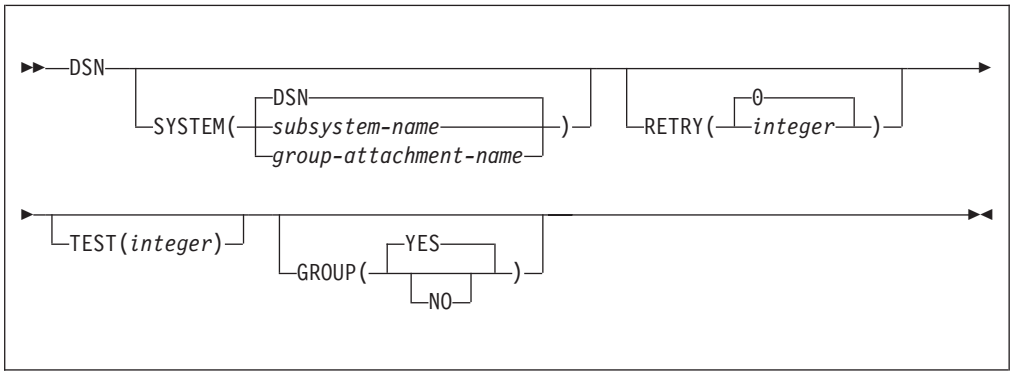
Data sharing scope: Member

Authorization

None is required for the DSN command, but authorization is required for most subcommands.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

None of the following options are required.

SYSTEM

(*subsystem-name***)**
Specifies the name of the DB2 subsystem.

(*group-attachment-name***)**
Specifies the group attachment name of the data sharing group.

The **default** is **SYSTEM(DSN).** This value can be modified during DB2 installation.

RETRY(*integer***)**
Specifies the number (integer) of additional times connection to the DB2 subsystem should be attempted if DB2 is not up or the maximum number of batch connections has been reached when DSN is issued. Retries occur at 30-second intervals.

The **default** is **RETRY(0)**. The maximum number of retries is 120.

TEST(*integer***)**
Specifies the last two digits (integer) of the module name in order to trace a single DSN module. Specify a number greater than 100 to trace all DSN modules. DSN trace information messages are written to the TSO SYSTSPRT DD statement, and optionally, to the DSNTRACE DD statement.

GROUP

(YES) Specify to consider group attach processing when the specified system is not active.

(NO) Specifies that group attach processing is not considered.

Usage notes

Beginning a DSN session: Issue the DSN command to begin a DSN session, which allows you to enter DSN subcommands. The following rules govern the session:

- In foreground operation, you are prompted for input by the prompt string DSN at the terminal. In background mode, your input is read from the SYSTSIN data set.

- Except for delimited table names in the DCLGEN command, input in lowercase letters is changed to uppercase.
- If duplicate keywords of any subcommand are specified, only the last of these keywords is processed. For example, if both MEMBER(*dbrm-member-name1*) and MEMBER(*dbrm-member-name2*) are specified with BIND PLAN, DB2 receives only the latter, MEMBER(*dbrm-member-name2*).
- If ATTENTION (PA1) is pressed during a DSN session, and PROMPT is specified in the TSO user profile, message DSNE005 appears: `EXECUTION IS INTERRUPTED, ENTER C TO CANCEL, OR ANY OTHER REPLY TO RESUME THE subcommand SUBCOMMAND.`

  If you enter C, the current subcommand is canceled and the current DB2 connection terminates; a new one is established, and another DSN prompt appears. Any other reply, except ATTENTION, causes the current subcommand to continue from the point at which it was interrupted.

  If a DSN session is started from a CLIST, or a CLIST is executed under DSN, CONTROL PROMPT must be specified in the CLIST in order to receive message DSNE005.
- After a command is processed during a DSN session, you are prompted for input. That cycle continues until you end the session.
- You can end the session by doing one of the following:
  - Issue the END subcommand. Control is passed to TSO.
  - Press ATTENTION and respond to the message by pressing ATTENTION again.
  - Issue another DSN command. The old session ends and a new one begins.

DSN return code processing: At the end of a DSN session, register 15 contains the highest value used by any DSN subcommand in the session or by any program run using the RUN subcommand. Your run-time environment might format that value as a return code. The value does not, however, originate in DSN.

Examples

Example 1: Start a DSN session. If the attempt to connect to DB2 fails, up to five retries (at 30 second intervals) will be made.

```
DSN SYSTEM (DB2) RETRY (5)
```

Example 2: Start a DSN session, run a program, and then end the session and return to TSO.

```
TSO prompt :  READY
USER enters:  DSN SYS (SSTR)
DSN prompt :  DSN
USER enters:  RUN PROGRAM (MYPROG)
DSN prompt :  DSN
USER enters:  END
TSO prompt :  READY
```

Chapter 35. DSNC (CICS attachment facility)

The CICS attachment facility DSNC command allows you to enter DB2 commands from CICS.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage note" on page 222
- "Example" on page 222

Environment

This command can be issued only from a CICS terminal.

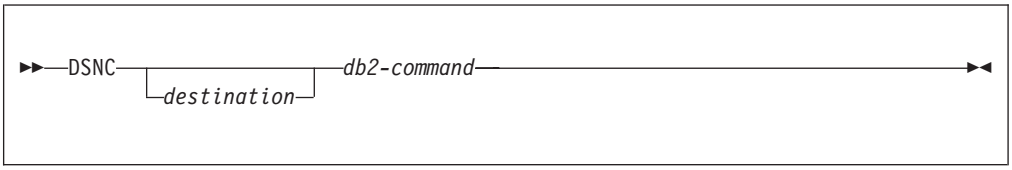
Data sharing scope: Member

Authorization

This command requires the appropriate level of CICS authority, as described in the appropriate *CICS Transaction Server for z/OS CICS-RACF Security Guide* or *CICS Transaction Server for z/OS Operations and Utilities Guide.*

Entering the DSNC command requires no privileges from DB2 security. For a description of the privileges required to issue a DB2 command using the DSNC command, see the command's description.

Syntax



Option descriptions

destination
Identifies another terminal to receive display information. It must be a valid terminal that is defined to CICS and supported by CICS basic mapping support (BMS).

db2-command
Specifies the exact DB2 command that you want to enter from a CICS terminal. It must be preceded by a hyphen.

© Copyright IBM Corp. 1983, 2008

Usage note

Screen scrolling: The CICS SIT table keyword SKR*xxxx* can be used to support the scrolling of DSNC DB2 commands from your terminal. For further information regarding the SIT keywords and parameters, see *CICS Transaction Server for z/OS System Definition Guide.*

Example

Example: Issue the DB2 command DISPLAY THREAD from a CICS terminal.

```
DSNC -DISPLAY THREAD
```

Chapter 36. DSNC DISCONNECT (CICS attachment facility)

The CICS attachment facility command DSNC DISCONNECT disconnects threads.

The command provides manual control to release resources being shared by normal transactions so that special purpose processes, such as utilities, can have exclusive access to the resources.

Abbreviation: DSNC DISC

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option description"
- "Usage notes"
- "Example" on page 224

Environment

This command can be issued only from a CICS terminal.

Data sharing scope: Member

Authorization

This command requires an appropriate level of CICS authority, as described in the appropriate *CICS Transaction Server for z/OS CICS-RACF Security Guide*.

Syntax

Option description

plan-name
Specifies a valid application plan.

Usage notes

Preventing creation of threads: The command DSNC DISCONNECT does not prevent threads from being created on behalf of transactions. The command only causes currently connected threads to be terminated as soon as they are not being used by a transaction. To interrupt a transaction and cancel a thread faster, you can use the DB2 command CANCEL THREAD. For details, see Chapter 16, "-CANCEL THREAD (DB2)," on page 97.

© Copyright IBM Corp. 1983, 2008

You can stop the transactions associated with a particular plan ID in CICS with the MAXACTIVE setting for TRANCLASS. This prevents new instances of the transaction from causing a re-creation of a thread.

Alternative for protected threads: You might want to deallocate a plan for rebinding or for running a utility against the database. If you are using a protected thread, use DSNC MODIFY rather than DSNC DISCONNECT. Modify the THRDA value of the plan to zero to send all the threads to the pool. The protected thread will terminate on its own within 60 seconds and DISCONNECT is unnecessary.

Example

Disconnect active threads for PLAN1.

```
DSNC DISCONNECT PLAN1
```

Chapter 37. DSNC DISPLAY (CICS attachment facility)

The CICS attachment facility command DSNC DISPLAY displays information on CICS transactions accessing DB2 data, or statistical information associated with DB2ENTRYs and the DB2CONN.

Abbreviation: DSNC DISP

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage notes" on page 226
- "Output" on page 226
- "Examples" on page 228

Environment

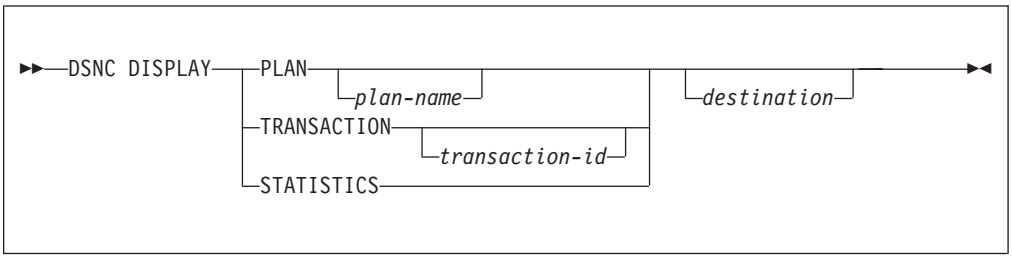
This command can be issued only from a CICS terminal.

Data sharing scope: Member

Authorization

This command requires an appropriate level of CICS authority, as described in the *CICS Transaction Server for z/OS CICS-RACF Security Guide* or *CICS Transaction Server for z/OS Operations and Utilities Guide.*

Syntax



Option descriptions

PLAN *plan-name*
Displays information about transactions by plan name.

plan-name is a valid plan name for which information is displayed.

Default: If you do not specify *plan-name* (or if you specify an asterisk, *), information is displayed for all active transactions.

© Copyright IBM Corp. 1983, 2008

TRANSACTION *transaction-id*
Displays information about transactions by transaction ID.

Abbreviation: TRAN

transaction-id is a valid transaction ID for which information is displayed.

Default: If you do not specify a transaction ID, information is displayed for all active transactions.

STATISTICS
Displays one line of the statistical counters that are associated with each DB2ENTRY. The counters correspond to the usage of the available connections of the CICS attachment facility to DB2.

Abbreviation: STAT

If you issue this command from CICS while the CICS attachment facility is active but the DB2 subsystem is not, CICS produces a statistics display with no obvious indication that the DB2 subsystem is not operational. Message DFHDB2037 appears in the CICS message log to indicate that the attachment facility is waiting for DB2 to start.

For a description of the output produced by this parameter, see Part 4 (Volume 1) of *DB2 Administration Guide*.

destination
Specifies the identifier of another terminal that is to receive the requested display information. It must be a valid terminal that is defined to CICS and supported by CICS basic mapping support (BMS).

Usage notes

Entering parameters: Because the optional destination is sometimes preceded by an optional plan name or transaction ID in the command, each parameter must be unique and separately identifiable as either a name or a terminal identifier. If only one parameter is entered, it is first checked to see whether it is a plan name or a transaction ID, and it is then checked as a destination. To use a character string that is both a plan name or transaction ID and also a valid terminal identifier, you must use both the name and destination parameters to display the desired information at the desired terminal.

Acknowledging display information sent to an alternative destination: When an alternative destination is specified to receive the requested display information, the following message is sent to the requesting terminal:

```
DFHDB2032  THE DISPLAY COMMAND IS COMPLETE
```

Output

For each created thread, the output for the DSNC DISPLAY (PLAN or TRANSACTION) command, as seen in ″Example 2,″ on page on page 228, displays the following information:

DB2ENTRY The name of the DB2ENTRY, *POOL for the pool, or *COMMAND for DSNC command calls.

S The status field. A status of A indicates the thread is active within a unit of work. A status of I indicates that a protected thread is waiting for work.

PLAN The plan that is associated with the thread. Command threads have no plan.

PRI –AUTH The primary authorization ID for the thread.

SEC –AUTH The secondary authorization ID (if any) for the thread.

CORRELATION
The 12-byte thread correlation ID in the form of *eeeetttnnnn*, where *eeee* is either COMD, POOL, or ENTR, indicating a command, pool, or DB2ENTRY thread; *tttt* is the transaction ID; *nnnn* is a unique number.

If the thread is active within a unit of work, its CICS transaction name (TRAN), task number (TASK), and CICS local unit of work ID (UOW-ID) are also displayed.

The output of a DSNC DISPLAY STATISTICS command, as seen in "Example 4" on page 228, displays the following information:

DB2ENTRY The name of the DB2ENTRY, *COMMAND for DSNC command calls, or *POOL for pool statistics.

PLAN The plan name that is associated with this entry. Eight asterisks in this field indicate that this transaction is using dynamic plan allocation. The command processor transaction DSNC does not have a plan associated with it because it uses a command processor.

CALLS The total number of SQL statements that are issued by transactions that are associated with this entry.

AUTHS The total number of sign-on invocations for transactions associated with this entry. A sign-on does not indicate whether a new thread is created or an existing thread is reused. If the thread is reused, a sign-on occurs only if the authorization ID or transaction ID has changed.

W/P The number of times that all available threads for this entry were busy. This value depends on the value of THREADWAIT for the entry.

An overflow to the pool is displayed in the transaction statistics only and is not reflected in the pool statistics.

If THREADWAIT is set to YES, the output reflects the number of times that the thread had to wait. If the number of started tasks has reached THREADLIMIT, the output also reflects the number of times the thread could not attach a new subtask.

The only time W/P is updated for the pool is when a transaction had to wait for a pool thread and a new subtask could not be attached for the pool. The W/P statistic is useful for determining if a sufficient number of threads are defined for the entry.

HIGH The maximum number of threads that are required by transactions that are associated with this entry at any time since the connection was started. It provides a basis for setting the maximum number of threads for the entry. For releases of CICS Transaction Server before release 1.2, this number includes the transactions that were forced to wait or diverted to the pool. For release 1.2 or later, the HIGH keyword is associated only with the threads that are actually created on the entry.

ABORTS The total number of units of recovery that were rolled back. It includes both abends and SYNCPOINT ROLLBACKS, including SYNCPOINT ROLLBACKS generated by -911 SQL codes.

COMMITS One of the following two fields increments each time a DB2 transaction that is associated with this entry has a real or implied (such as EOT) syncpoint. Units of recovery that do not process SQL calls are not reflected here.

1-PHASE The total number of single phase commits for transactions that are associated with this entry. This total does not include any two-phase commits (see the explanation for 2-PHASE). This total does include read-only commits and single-phase commits for units of recovery that have performed updates. A two-phase commit is needed only when CICS is the recovery coordinator for more than one resource manager.

2-PHASE The total number of two-phase commits for transactions that are associated with this entry. This number does not include one-phase commit transactions.

Examples

Example 1: This command displays information about all active threads. The display information is to be sent to another terminal designated as MTO2.

```
DSNC DISP PLAN * MTO2
```

Example 2: This example shows the output for the DSNC DISPLAY (PLAN or TRANSACTION) command.

```
DFHDB2013 07/09/98 15:26:47 IYK4Z2G1 DISPLAY REPORT FOLLOWS FOR THREADS
                                     ACCESSING DB2 DB3A

DB2ENTRY S PLAN     PRI-AUTH SEC-AUTH CORRELATION  TRAN TASK  UOW-ID
*POOL    A TESTC05  JTILLI1           POOLXC050001 XC05 01208 AEEEC03-1ACDCE00
XC06     A TESTC06  JTILLI1           ENTRXC060003 XC06 01215 AEEEC04-2F8EFE01
XP05     A TESTP05  JTILLI1           ENTRXP050002 XP05 01209 AEEEC03-35230C00
XP05     I TESTP05  JTILLI1           ENTRXP050004
DFHDB2020 07/09/98 15:26:47 IYK4Z2G1 THE DISPLAY COMMAND IS COMPLETE.
```

Example 3: This command displays statistical counters that are associated with DB2ENTRY.

```
DSNC DISP STAT
```

Example 4: This example shows the output for the DSNC DISPLAY STATISTICS command:

```
DFHDB2014 07/09/98 14:35:45 IYK4Z2G1 STATISTICS REPORT FOLLOWS
                                                          -----COMMITS-----
DB2ENTRY  PLAN           CALLS   AUTHS    W/P  HIGH   ABORTS   1-PHASE   2-PHASE
*COMMAND                   1       1       1    1       0         0         0
*POOL     POOL             0       0       0    0       0         0         0
XC01      DSNXC01         22       1      11    2       0         7         5
XC02      DSNXC02          0       0       0    0       0         0         0
XA81      DSNA81           0       0       0    0       0         0         0
XCD4      DSNCED4          0       0       0    0       0         0         0
XP03      DSNTP03          1       1       0    1       0         1         0
XA20      DSNTA20          1       1       0    1       0         0         1
XA88      ********         0       0       0    0       0         0         0
DFHDB2020 07/09/98 15:45:27 IYKA4z2G1   THE DISPLAY COMMAND IS COMPLETE
```

Chapter 38. DSNC MODIFY (CICS attachment facility)

The CICS attachment facility command DSNC MODIFY modifies:

- The message queue destination of the DB2CONN
- The maximum active thread value for the pool, for DSNC commands, or for DB2ENTRY

Abbreviation: DSNC MODI

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage notes" on page 230
- "Examples" on page 230

Environment

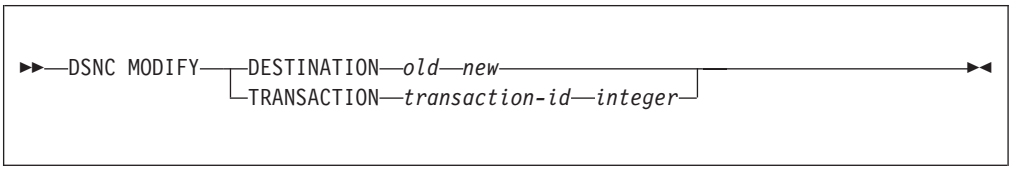
This command can be issued only from a CICS terminal.

Data sharing scope: Member

Authorization

This command requires an appropriate level of CICS authority, as described in the appropriate *CICS Transaction Server for z/OS CICS-RACF Security Guide* or *CICS Transaction Server for z/OS Operations and Utilities Guide.*

Syntax



Option descriptions

DESTINATION

Specifies that the MSGQUEUE parameter of the DB2CONN is to be changed, replacing the *old* destination ID with the *new* destination ID.

Abbreviation: DEST

old

Specifies any destination ID that is currently active in the MSGQUEUE of the DB2CONN.

© Copyright IBM Corp. 1983, 2008

new
Specifies a new destination identifier. CICS verifies the new destination to ensure that it is an existing transient data entry in the destination control table.

TRANSACTION
Specifies that the maximum active thread value that is associated with the specified transaction or group is to be modified.

Abbreviation: TRAN

transaction-id
Specifies a valid transaction identifier.

To change the maximum active thread value, use one of the following transaction IDs:

- For the pool: CEPL
- For command threads: DSNC
- For DB2ENTRY: the ID of any transaction that is defined to use DB2ENTRY

integer
Specifies a new maximum value.

Usage notes

Protected threads: If you increase the active thread value by using the command `DSNC MODIFY TRANSACTION`, the attributes of the DB2ENTRY are used.

Issuing DSNC MODIFY TRANSACTION to increase the total number of threads that are permitted allows creation of unprotected threads. For example, assume PROTECTNUM(2) and THREADLIM(2). If the total number of permitted threads increases, the additional threads are unprotected.

The command DSNC MODIFY TRANSACTION can also allow creation of protected threads. If PROTECTNUM(2) and THREADLIM(2) and you modify the thread limit to 1, one of the protected threads is eliminated. If the thread limit is then modified back to 2, the thread that is re-created is protected.

TRANSACTION thread limit: The lowest possible value is zero.

Examples

Example 1: Change the specification of the MSGQUEUE parameter in the DB2CONN from MTO1 to MTO2.

```
DSNC MODIFY DESTINATION MTO1 MTO2
```

Example 2: Change the pool thread limit to 12.

```
DSNC MODIFY TRANSACTION CEPL 12
```

Example 3: Change the command thread limit to 3.

```
DSNC MODIFY TRANSACTION DSNC 3
```

Example 4: Change the thread limit of the DB2ENTRY that is used by the transaction XP05 to 8.

```
DSNC MODIFY TRANSACTION XP05 8
```

Chapter 39. DSNC STOP (CICS attachment facility)

The CICS attachment facility command DSNC STOP stops the attachment facility.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage notes" on page 232
- "Example" on page 232

Environment

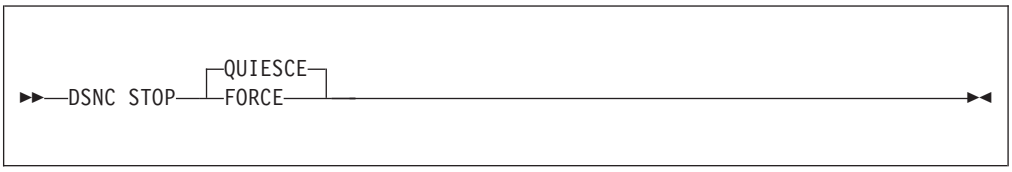
This command can be issued only from a CICS terminal.

Data sharing scope: Member

Authorization

This command requires an appropriate level of CICS authority, as described in the appropriate *CICS Transaction Server for z/OS CICS-RACF Security Guide* or *CICS Transaction Server for z/OS Operations and Utilities Guide.*

Syntax



Option descriptions

QUIESCE
Specifies that the CICS attachment facility is to be stopped after CICS transactions that are currently running terminate.

Abbreviation: Q

FORCE
Specifies that the CICS attachment facility is to be stopped immediately by forcing disconnection with DB2, regardless of any transactions that are running.

© Copyright IBM Corp. 1983, 2008

Usage notes

Requirements for restarting: Using FORCE can leave threads in an indoubt situation. Restarting requires reconnection of CICS and DB2 to resolve any indoubt situations. In a data sharing environment, resolution of indoubt situations requires that the CICS be reconnected to the same DB2 member.

Output destinations: Output from the command DSNC STOP is sent to the requesting terminal, which remains locked until shutdown is complete.

Example

Stop the CICS attachment facility.

```
DSNC STOP FORCE
```

Chapter 40. DSNC STRT (CICS attachment facility)

The DSNC STRT command starts the CICS attachment facility, which allows CICS application programs to access DB2 databases.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option description"
- "Usage note"
- "Examples" on page 234

Environment

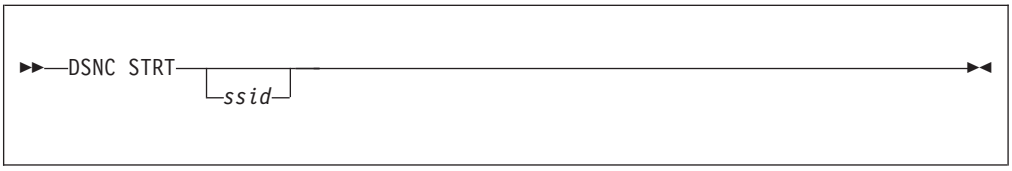
This command can be issued only from a CICS terminal.

Data sharing scope: Member

Authorization

This command requires an appropriate level of CICS authority, as described in the appropriate *CICS Transaction Server for z/OS CICS-RACF Security Guide* or *CICS Transaction Server for z/OS Operations and Utilities Guide.*

Syntax



Option description

ssid

Specifies the subsystem ID (SSID) that is to override the ID that is specified in the CICS DB2CONN.

Default: The DB2ID that is specified in the last installed DB2CONN. If the DB2CONN contains a blank DB2ID, the default is the SSID that is specified in the CICS INTIPARM parameter.

Usage note

Output destinations: Output from the DSNC START command is sent to the requesting terminal. If no DB2CONN is installed when you issue DSNC STRT, error message DFHDB2031 is sent to the terminal.

© Copyright IBM Corp. 1983, 2008

Examples

Example 1: Start the CICS attachment facility.

```
DSNC STRT
```

Example 2: Start the CICS attachment facility. Override the SSID that is specified in the DB2CONN with DB2P.

```
DSNC STRT DB2P
```

Example 3: Start the CICS attachment facility. Use SSID DBA1.

```
DSNC STRT DBA1
```

Chapter 41. DSNH (TSO CLIST)

The DSNH command procedure (a TSO CLIST) is a powerful yet easy method of preparing an application program for execution. By issuing a single command, you can select numerous options required for the preparation of an application and execute it under TSO.

DSNH processing is a sequential process that can include any of the actions listed in Table 20 referred to by the two-letter step name:

Table 20. DSNH actions and the corresponding step names

For invoking the...	Use step name
PL/I macro processor	MP
DB2 precompiler	PC
CICS command language translator	TR
DSN BIND PLAN subcommand for binding a plan	BI
DSN BIND PACKAGE subcommand for binding a package	BP
Compiler or assembler for your program	CO
A C compiler prelink utility for including compile-time parameters	PL
Link-editor to produce an executable load module	LE
DSN RUN subcommand to execute the program	RU
Note: The step names are used in the heading of Table 22 on page 237.	

Individual steps or a sequence of steps can be performed, and you can end the process at any point you choose. Any steps in the process that are skipped must have previously been completed successfully by DSNH. For guidance in preparing an application program to run, refer to *DB2 Application Programming and SQL Guide*. Refer to Table 1 on page 21 for a description of the DSN BIND subcommands.

The following topics provide additional information:

- "Environment" on page 236
- "Authorization" on page 236
- "Syntax" on page 236
- "Summary of DSNH CLIST parameters" on page 236
- "General parameter descriptions" on page 240
- "DSNH/DSN subcommand summary" on page 254
- "DSNH CLIST/BIND PLAN subcommand comparison" on page 254
- "DSNH CLIST/BIND PACKAGE subcommand comparison" on page 257
- "Usage notes" on page 260
- "Examples" on page 261

© Copyright IBM Corp. 1983, 2008

Environment

The DSNH CLIST can run in TSO foreground or in batch under the TSO terminal monitor program. DB2I uses the DSNH CLIST on the precompiler panel to control program preparation. You can pass DSNH parameters from DB2I panels on the ″Other options″ lines.

Data sharing scope: Member

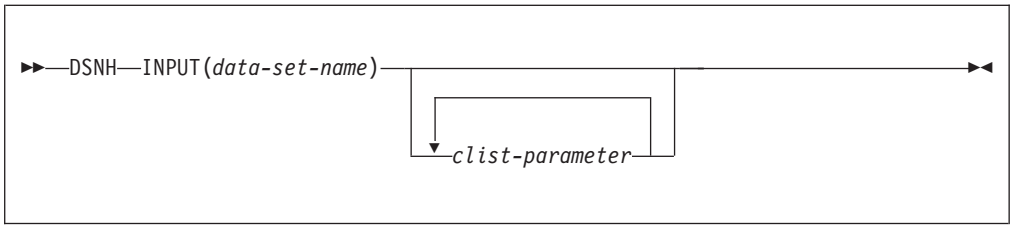
Authorization

See Chapter 13, “BIND PACKAGE (DSN),” on page 49 for a description of the privileges necessary to bind a package.

See Chapter 14, “BIND PLAN (DSN),” on page 57 for a description of the privileges necessary to bind a plan.

See Chapter 61, “RUN (DSN),” on page 327 for a description of the privileges necessary to run a plan.

Syntax



Summary of DSNH CLIST parameters

The CLIST parameters provide the processing options for each step; specify them when you execute DSNH. Some parameters are used for more than one step, as indicated in Table 22 on page 237. Table 22 on page 237 shows where each parameter is used, using the following notation:

- Y in any cell shows that the option listed at the beginning of the row is used in the step whose name appears at the top of the column.
- * in any cell indicates that the option listed at the beginning of the row is used in *another* step which affects the step whose name appears at the top of the column.

Notation of CLIST parameters for the BIND PLAN and BIND PACKAGE steps: Many parameters of BIND PLAN and of BIND PACKAGE provide the same function and are spelled alike. CLIST parameters for BIND PLAN and BIND PACKAGE are differentiated from general parameters and from each other by prefixes. A parameter name prefixed by the letter B applies to the BIND PLAN subcommand; a parameter name prefixed by the letter P applies to BIND PACKAGE. Table 21 on page 237 shows the possible variations for a single parameter name.

Table 21. DSNH CLIST prefixing rules

Parameter value	Function or subcommand	Example
parameter	If no prefix is specified, the parameter applies to a single function or subcommand.	`DBRMLIB`
B/*parameter*	The prefix B is used to indicate that this variation of the parameter applies only to the BIND PLAN step.	`B/DBRMLIB`
P/*parameter*	The prefix P is used to indicate that this variation of the parameter applies only to the BIND PACKAGE step.	`P/DBRMLIB`

In Table 22, a prefix is separated from the DB2 parameter name by a slash (/). Refer to Table 20 on page 235 for an explanation of the two-letter step names.

Table 22. Summary of DSNH CLIST parameters

OPTIONS	MP	PC	TR	BI	BP	CO	PL	LE	RU
ACQUIRE				Y					*
P/ACTION				Y	Y				
ASMLIB						Y			
ASMLOAD						Y			
P/BDMEM				Y	Y				
P/BIND				Y	Y				
P/BLIB				Y					
P/BnLIB				Y					
P/BMEM				Y	Y				
CACHESIZE				Y					
CCLINK							Y		
CCLLIB								Y	
CCLOAD						Y			
CCMSGS						Y	Y		
CCOLIB							Y		
CCPLIB								Y	
CCPMSGS							Y		
CCSID		Y							
CCSLIB						Y			
P/CICS				Y	Y				
CICSCOB			Y					Y	
CICSLLIB			Y					Y	
CICSOPT			Y						
CICSPRE			Y					Y	
CICSPLIB			Y					Y	
CICSVER			Y					Y	
CICSXLAT			Y						
CLIB	Y					Y			
CnLIB	Y					Y			
COBICOMP								Y	
COBILINK								Y	
COBIPLNK							Y		

Table 22. Summary of DSNH CLIST parameters (continued)

OPTIONS	MP	PC	TR	BI	BP	CO	PL	LE	RU
COBIPMSG							Y		
COBLIB								Y	
COBLOAD						Y			
COBSOM							Y		
COB2CICS								Y	
COB2LIB								Y	
COB2LOAD						Y			
COMPILE						Y			
CONNECT		Y							
CONTROL	Y		Y	*		Y		Y	Y
COPTION	Y					Y			
COPY					Y				
COPYVER					Y				
CPPCLASS							Y		
CPPCLINK							Y		
CPPCLLIB								Y	
CPPCSLIB						Y			
CPPLLIB							Y		
CPPPMSGS							Y		
CPPSLIB						Y			
CPPUTIL						Y			
CURRENTDATA				Y	Y				
CURRENTSERVER				Y					
DATE		Y							
P/DBPROTOCOL				Y	Y				
P/B/DBRMLIB		Y		Y	Y				
DECARTH		Y							
DECIMAL		Y				*			
P/DEFER				Y	Y				
P/DEGREE				Y	Y				
DELIMIT		Y	Y			Y			
P/DISABLE				Y	Y				
DISCONNECT				Y	-				
P/DLIBATCH				Y	Y				
P/DYNAMICRULES				Y	Y				
P/ENABLE				Y	Y				
ENTRY								Y	
EXPLAIN				Y	Y				
P/FLAG	Y	Y	Y	Y	Y	Y			
FORTLIB								Y	
FORTLOAD						Y			
GRAPHIC		Y							
HOST	Y	Y	Y	*		Y		Y	Y
P/IMSBMP				Y	Y				
P/IMSMPP				Y	Y				

Table 22. Summary of DSNH CLIST parameters (continued)

OPTIONS	MP	PC	TR	BI	BP	CO	PL	LE	RU
IMSPRE								Y	
INPUT	Y	Y	*	*		Y		Y	Y
P/ISOLATION				Y	Y				
P/KEEPDYNAMIC				Y	Y				
LINECOUNT	Y	Y	Y			Y			
LINK								Y	
LLIB								Y	
LnLIB								Y	
LOAD								Y	Y
LOPTION								Y	
MACRO	Y		Y						
NEWFUN		Y							
NOFOR	Y								
P/NODEFER				Y	Y				
P/OPTHINT				Y	Y				
OPTIONS		Y	Y					Y	Y
OUTNAME	Y	Y				Y		Y	Y
P/OWNER				Y	Y				
PACKAGE					Y				
PARMS									Y
PASS		Y							
P/PATH				Y	Y				
PCLOAD		Y							
PKLIST				Y					
PLAN				Y					Y
PLIB		Y							
PnLIB		Y							
PLI2LIB								Y	
PLILIB								Y	
PLILOAD	Y					Y			
PLIPLNK								Y	
PLIPMSG								Y	
POPTION							Y		
PRECOMP		Y							
PRELINK							Y		
PRINT	Y	Y	Y			Y		Y	
PSECSPAC	Y	Y	Y			Y			Y
PSPACE	Y	Y	Y			Y			Y
P/QUALIFIER				Y	Y				
RCTERM	Y	Y	Y	Y	Y	Y	Y		Y
P/RELEASE				Y	Y				
REMOTE					Y				
P/REOPT				Y	Y				
REPLVER					Y				
RETAIN				Y					

Table 22. Summary of DSNH CLIST parameters (continued)

OPTIONS	MP	PC	TR	BI	BP	CO	PL	LE	RU
RUN		Y	Y					Y	Y
RUNIN									Y
RUNOUT									Y
SOMDLLI						Y	Y		
SOURCE	Y	Y	Y			Y			
SPACEUN	Y	Y	Y			Y			Y
SQL		Y							
SQLDELIM		Y							
SQLERROR					Y				
SQLRULES				Y					
STDSQL		Y							
SUFFIX	Y	Y							
SYSTEM				*	*				Y
TERM	Y	Y				Y		Y	
TIME		Y							
P/VALIDATE				Y	Y				
VERSION		Y							
WORKUNIT	Y	Y	Y			Y			
WSECSPAC	Y	Y	Y			Y			
WSPACE	Y	Y	Y			Y			
XLIB								Y	
XREF	Y	Y				Y		Y	

General parameter descriptions

Due to similarities in name and function, the CLIST parameters for BIND PLAN and BIND PACKAGE are described separately from the parameters in Table 23. For a summary of:

- BIND PLAN parameters, refer to Table 24 on page 254
- BIND PACKAGE parameters, refer to Table 25 on page 257.

Also see "DSNH/DSN subcommand summary" on page 254 for a description of conventions used in those tables.

The only parameter that is required on the DSNH statement is INPUT; the others are optional. In Table 23 on page 241:

- Parameter values must be enclosed between parentheses.
- Parameter values need not be enclosed between apostrophes, except in either of the following cases:
  - If the value is a list of tokens with separators, the value must be enclosed between apostrophes.
  - If the value is a data set name, your user identifier is added as a prefix. To avoid the prefix, enclose the data set name between sets of three apostrophes.
- Most parameter values that are data set names (*dsname*) cannot include member names. Exceptions are noted in the parameter descriptions.
- Underlined values are defaults. Default names can be changed to names specific to your site when DB2 is installed.

Table 23. General DSNH CLIST parameters

Parameter	Value	Comments
ASMLIB	*dsname*	Specifies a data set to be used as the standard MACLIB for High Level Assembler. The **default** is "'SYS1.MACLIB'".
ASMLOAD	*dsname*	Specifies a data set that contains the High Level Assembler load module. *dsname* can include a member name. The **default** is "'SYS1.LINKLIB(ASMA90)'".
CCLINK	*dsname*	Specifies a data set that contains the IBM Language Environment® prelink editor utility invocation load module that is to be used for preparing C programs. *dsname* can include a member name. The **default** is "'CEE.SCEERUN(EDCPRLK)'".
CCLLIB	*dsname*	Specifies a data set that contains the linkage editor include modules for the C compiler routines. The **default** is "'CEE.SCEELKED'".
CCLOAD	*dsname*	Specifies a data set that contains the C compiler invocation load module. *dsname* can include a member name. The **default** is "'CBC.SCCNCMP(CCNDRVR)'".
CCMSGS	*dsname*	Specifies a data set that contains the C compiler messages. This data set is required only for C/370. *dsname* can include a member name. The **default** is "'EDC.V1R2M0.SEDCDMSG(EDCMSGE)'".
CCSID	*integer*	Specifies the CCSID for source SQL statements as the integer. See the description of precompiler options in *DB2 Application Programming and SQL Guide* for more information about the CCSID parameter.
CCOLIB	NONE *dsname*	Specifies that the data set that contains C object modules is included during the execution of the prelink utility step.
CCPLIB	NONE *dsname*	Specifies a data set that contains include modules for PL/I routines. This parameter is used only for IBM C/370 Version 2 or earlier.
CCPMSGS	*dsname*	Specifies a data set that contains the message library that is to be used by the IBM prelink editor when preparing C programs. The **default** is "'CEE.SCEEMSGP(EDCPMSGE)'".
CCSLIB	*dsname*	Specifies a data set that contains the C compiler headers. The **default** is "'CEE.SCEEH.H'".
CICSOPT	NONE *option-list*	Specifies a list of additional CICS translator options. See the appropriate CICS application programming reference for information about translator options. The **default**, NONE, specifies no additional options.

Table 23. General DSNH CLIST parameters (continued)

Parameter	Value	Comments
CICSPRE	*prefix*	Specifies the prefix for the CICS libraries. The library names are: *prefix*.LOADLIB for translators *prefix*.PL1LIBn for PL/I include *prefix*.COBLIB for COBOL include Leave this parameter blank to use CICSLLIB, CICSPLIB, CICSCOB. The **default** is blank.
CICSLLIB	*dsname*	Specifies the CICS load library. To use this library, leave the CICSPRE parameter blank. The **default** is set on installation panel DSNTIP3.
CICSPLIB	*dsname*	Specifies the CICS PL/I library. To use this library, leave the CICSPRE parameter blank. The **default** is set on installation panel DSNTIP3.
CICSCOB	*dsname*	Specifies the CICS COBOL library. To use this library, leave the CICSPRE parameter blank. The **default** is set on installation panel DSNTIP3.
CICSVER	21 31 33 41	Specifies the CICS release. This field is ignored because current releases of CICS do not require DSNH to handle release-specific considerations.
CICSXLAT	NO YES	Specifies whether to execute the CICS command translator. This parameter is effective only if you use RUN(CICS). You cannot use this parameter with the MARGINS option of the translator. The **default** is YES. The DB2I panel **default** is NO.
CLIB C*n*LIB	NONE *dsname*	Specifies a data set that contains host language source statements to be included by the compiler or assembler. The parameters C*n*LIB (where *n* can be 2, 3, or 4) are extensions of CLIB, which is used to simplify passing a list of data set names. Use the **default**, NONE, to specify no data set.
COBICOMP	*dsname*	Specifies the IBM COBOL data set that is required for compilation. The **default** is "'IGY.SIGYCOMP'".
COBILINK	*dsname*	Specifies the IBM COBOL data set that is required for link edit. The **default** is "'CEE.SCEELKED'".
COBIPLNK	*dsname*	Specifies a data set that contains the IBM Environment prelink editor utility invocation load module that is to be used for preparing COBOL programs. *dsname* can include a member name. The **default** is "'CEE.SCEERUN(EDCPRLK)'".
COBIPMSG	*dsname*	Specifies a data set that contains the message library that is to be used by the IBM prelink editor when preparing COBOL programs. *dsname* can include a member name. The **default** is "'CEE.SCEEMSGP(EDCPMSGE)'".

Table 23. General DSNH CLIST parameters (continued)

Parameter	Value	Comments
COBLIB	*dsname*	Specifies the linkage editor include library that is to be used for OS/VS COBOL routines. This parameter is obsolete.
COBLOAD	*dsname*	Specifies a data set that contains the OS/VS COBOL compiler load module. This parameter is obsolete.
COBSOM	*dsname*	Specifies the IBM System Object Model® (SOM®) data set that is required for access to SOM objects. This parameter is obsolete.
COB2CICS	*dsname*	Specifies the linkage editor include library that is to be used for VS COBOL II CICS routines. This parameter is obsolete.
COB2LIB	*dsname*	Specifies the linkage editor include library that is to be used for the VS COBOL II or COBOL/370 routines. This parameter is obsolete.
COB2LOAD	*dsname*	Specifies a data set that contains the VS COBOL II or COBOL/370 compiler load module. This parameter is obsolete.
COMPILE	YES NO	Specifies whether to execute the compiler or assembler if the precompile step is successful.
CONNECT	(1) (2)	Specifies whether a CONNECT SQL statement should be processed as a type 1 CONNECT or a type 2 CONNECT statement. The DSNH(TSO CLIST) command does not accept the CT(1) and CT(2) abbreviations for this precompiler option. The **default** is CONNECT(2).
CONTROL	NONE CONLIST LIST SYMLIST	Specify to help you trace the allocation of non-existent data sets. Use this parameter if you have a problem without an obvious cause. CONLIST displays CLIST commands after substitution for symbols and before command execution. LIST displays TSO commands after substitution for symbols and before command execution. SYMLIST displays all executable statements (TSO commands and CLIST statements) before substitution for symbols.
COPTION	NONE *string*	Specifies a list of compiler or assembler options. For more information, refer to the manual that describes the compiler or assembler options for the specific language you are using. For a list of restrictions on some options, see "COBOL Options" on page 260. NONE specifies no options.
CPPCLASS	*dsname*	Specifies the data set that contains C++ class libraries. The **default** is "'CBC.SCLBCPP"'.

Table 23. General DSNH CLIST parameters (continued)

Parameter	Value	Comments
CPPCLINK	*dsname*	Specifies the data set that contains the IBM Language Environment prelink editor utility invocation load module that is to be used for preparing C programs. The **default** is "'CEE.SCEERUN(EDCPRLK)'".
CPPCLLIB	*dsname*	Specifies the data set for the C linkage editor automatic call library that is used by the C++ compiler. The **default** is "'CEE.SCEELKED'".
CPPCSLIB	*dsname*	Specifies a data set that contains the C compiler headers that are used by the C++ compiler. The **default** is "'CEE.SCEEH.H'".
CPPLLIB	*dsname*	Specifies a data set that contains the C++ prelink automatic call library. The **default** is "'CEE.SCEECPP'".
CPPPMSGS	*dsname*	Specifies the data set that contains the message library that is to be used by the IBM prelink editor when preparing C++ programs. *dsname* can include a member name. The **default** is "'CEE.SCEEMSGP(EDCPMSGE)'".
CPPSLIB	*dsname*	Specifies the data set that contains C++ header files for class libraries. The **default** is "'CBC.SCLBH.HPP'".
CPPUTIL	*dsname*	Specifies the data set that contains procedures to set up and execute the C++ compiler. The **default** is "'CBC.SCCNUTL'".
DATE	ISO JIS USA EUR LOCAL	Specifies the format of date values that are to be returned, which overrides the format that is specified as the location default. The **default** is the value that is supplied when DB2 is installed, and is written in the data-only load module, DSNHDECP.

Table 23. General DSNH CLIST parameters (continued)

Parameter	Value	Comments
DBRMLIB	DEFAULT *dsname(member)* NONE	Specifies the partitioned data set, and an optional member name, that contains the DBRM library and member name that is used during the DB2 precompile step. Because you can specify individual DBRM member and library names during each individual phase, you must use the DBRMLIB parameter and associated prefixes to identify a specific phase. DBRMLIB specifies the DBRM library and member that is defined on the DBRMLIB DD statement during DB2 precompiler processing. DEFAULT indicates that the same DBRM library data set that is defined for the DB2 precompiler process (DBRMLIB(*parameter*)) is also used on the LIBRARY(*dsname*) subcommand keyword. If the precompiler DBRMLIB is not specified, the default generated DBRMLIB library that is based on the INPUT data set name is used. *dsname* is generated using the DSNH OUTNAME parameter value, or its default, TEMP, with the constant DBRM appended to the prefix; for example, *outname*.DBRM or TEMP.DBRM. *member* is obtained from the data set member name that is specified on the DSNH INPUT parameter or from the data set name as follows: Given INPUT(*outname*.DBRM(*dbrmmem*)): – *outname*.DBRM(*dbrmmem*) - If the member name is specified – *outname*.DBRM(*dbrm*) - If no member name is specified NONE indicates that no LIBRARY(*dsname*) subcommand keyword is specified on invocation.
DECARTH	DEFAULT 15 31	Specifies the maximum precision of decimal numbers. DEFAULT designates the value chosen, during installation, for the DECIMAL ARITHMETIC field on the APPLICATION PROGRAMMING DEFAULTS panel. A value of 15 specifies that decimal arithmetic operations on decimal values with precision 15 or less are performed in accordance with the existing rules for determining the precision and scale of the result. A value of 31 specifies that decimal arithmetic operations on decimal values with precision 15 to 31 are performed in accordance with new rules for determining the precision and scale of the result. DECARTH is ignored for Fortran.
DECIMAL	COMMA PERIOD	Specifies the decimal point indicator for decimal and floating point literals. DECIMAL is valid only for COBOL programs; PERIOD is forced for all other programs. COMMA makes the indicator a comma. PERIOD makes the indicator a period. The **default** is the value of the DECIMAL POINT field, set on the DB2 APPLICATION PROGRAMMING DEFAULTS panel during installation.

Table 23. General DSNH CLIST parameters (continued)

Parameter	Value	Comments
DELIMIT	DEFAULT APOST QUOTE	Specifies the APOST or QUOTE precompiler option to indicate the string delimiter that is used within host language statements. DELIMIT is effective only for COBOL programs; APOST is forced for all other programs. DEFAULT designates the value chosen during installation for the STRING DELIMITER field on the APPLICATION PROGRAMMING DEFAULTS panel. APOST specifies the apostrophe as the string delimiter for host language statements. QUOTE specifies a quotation mark as the string delimiter for host language statements.
ENTRY	*entry-name*	Specifies the entry point that is assigned by the linkage editor. The **default** depends on the host language and the value of RUN. • For the PL/I language, the ENTRY value default is: – NONE if the RUN value is CICS – PLISTART for any other RUN value. • For assembler language, the ENTRY value default is DLITASM if the RUN value is IMS. • For COBOL, the ENTRY value default is DLITCBL if the RUN value is IMS. • For any other language, the ENTRY value default is NONE (no specified entry point) for any RUN value.
FLAG	I C E W	Specifies the messages that you want to see. Use one of the following values to show messages of the corresponding types: **I** All informational, warning, error, and completion messages **W** Only warning, error, and completion messages **E** Only error and completion messages **C** Only completion messages
FORTLIB	*dsname*	Specifies the linkage editor include library that is to be used for Fortran routines. The **default** is '"SYS1.VSF2FORT"'.
FORTLOAD	*dsname*	Specifies a data set that contains the VS Fortran compiler load module. *dsname* can include a member name. The **default** is '"SYS1.VSF2VCOMP(FORTVS2)"'.
GRAPHIC	NONE NO YES	Specifies the value of the DSNHDECP MIXED option for the precompiler. NONE indicates that the default specified during installation is used. NO indicates that the data is **not** mixed DBCS. YES indicates that all character data can be mixed DBCS. GRAPHIC is ignored for C.

Table 23. General DSNH CLIST parameters (continued)

Parameter	Value	Comments
HOST	ASM C CPP IBMCOB FORTRAN PLI	Defines the host language within which SQL statements are embedded. COBOL and COB2 are also acceptable values but are obsolete. If your program is one of the following types of programs, you cannot use DB2I to prepare it: • A COBOL program that uses object-oriented extensions • A C++ program that uses object-oriented extensions and consists of more than one compilation unit The **default** is the value of the LANGUAGE DEFAULT field, set on the DB2 APPLICATION PROGRAMMING DEFAULTS panel during installation.
IMSPRE	*prefix*	Specifies the prefix for RESLIB, which is used for routines that are to be included by the linkage editor for IMS. The **default** is IMSVS.
INPUT	*dsname*	Specifies the data set that contains the host language source and SQL statements. *dsname* can include a member name.
LINECOUNT	*integer*	Specifies how many lines, including headings, are to be printed on each page of printed output. The **default** is 60.
LINK	YES NO	Specifies whether to execute the linkage editor after successful completion of compilation or assembly. YES indicates that the linkage editor is to be executed. The DSNHLI entry point from the precompiler is directed to the appropriate language interface module that is specified by the RUN parameter. NO indicates that linkage editor processing is to be bypassed.
LLIB L*n*LIB	NONE *dsname*	Specifies a data set that contains object or load modules that are to be included by the linkage editor. The parameters L*n*LIB (where *n* can be 2, 3, or 4) are extensions of LLIB, which is used to simplify passing a list of data set names. The LLIB and L*n*LIB libraries are concatenated with the XLIB library and the linkage editor include libraries for the specific host language. Object and load module libraries must not be mixed in this concatenation. Use the **default,** NONE, to specify no data set.
LOAD	*dsname*	Specifies a data set that is to contain the output from the linkage editor (the load module). *dsname* can include a member name. The **default** is RUNLIB.LOAD.
LOPTION	NONE *string*	Specifies a list of linkage editor options. For information about the options you can use, see the appropriate z/OS publication. Use the **default,** NONE, to give no options.

Table 23. General DSNH CLIST parameters (continued)

Parameter	Value	Comments
MACRO	YES NO	Specifies whether the macro preprocessor is to be executed before the precompilation of a PL/I program. If the PL/I macro processor is used, the PL/I *PROCESS statement must not be used to pass options to the PL/I compiler. The COPTION parameter of the DSNH command can be used to pass the needed options to the PL/I compiler.
NEWFUN	NO YES	Specifies whether to allow syntax for functions that DB2 Version 8 introduces. See the description of precompiler options in *DB2 Application Programming and SQL Guide* for more information.
NOFOR	NO YES	Specifies whether all FOR UPDATE OF clauses in static SQL statements are optional. When you specify NOFOR(YES), the FOR UPDATE OF clause is optional. Positioned updates can be made to any columns that the user has authority to update. When you specify NOFOR(NO), any query that appears in a DECLARE CURSOR statement must contain a FOR UPDATE OF clause if the cursor is used for positional updates. The clause must designate all the columns that the cursor can update. The option is implied when the STDSQL(YES) option is in effect.
OPTIONS	NO YES	Specifies whether to print the options that are used when executing the precompiler or the CICS command translator with the output listing.
OUTNAME	TEMP *string*	Specifies the prefix that is used to form intermediate data set names. *string* must not be enclosed between apostrophes and must not have the same initial character as the *dsname* for INPUT. It cannot contain special characters.
PARMS	NONE *string*	Specifies a parameter string that is to be passed to the compiled program during its execution. This parameter is valid only if the run-time execution environment requested is TSO. If CAF is specified as the run-time execution environment, this parameter is ignored. Use the **default,** NONE, to pass no parameter string.
PASS	ONE or 1 TWO or 2	Specifies how many passes the precompiler is to use. One pass saves processing time, but requires that declarations of host variables in the program precede any reference to those variables. PASS has no effect for COBOL or Fortran; ONE is forced. The **default** is ONE or 1 for PL/I and C. The **default** is TWO or 2 for assembler.
PCLOAD	*dsname*	Specifies the precompiler load module. *dsname* can include a member name. The **default** is '*(DSNHPC)'.

Table 23. General DSNH CLIST parameters (continued)

Parameter	Value	Comments
PLAN	*plan-name*	Specifies the application plan that is created by the bind process. The **default** plan name is the first of the following available choices defined in the INPUT data set: • DBRM member name • Leftmost qualifier *plan-name* must not be DEFAULT. If no name is found, a plan is not created.
PLIB P*n*LIB	NONE *dsname*	Specifies the data set that contains host language source or SQL statements included by the SQL INCLUDE statement during precompilation. The parameters P*n*LIB (where *n* can be 2, 3, or 4) are extensions of PLIB, which is used to simplify passing a list of data set names. Use NONE to specify no data set.
PLI2LIB	*dsname*	Specifies the linkage editor common library that is used for PL/I routines. This parameter is obsolete.
PLILIB	*dsname*	Specifies the linkage editor base library that is used for PL/I routines. The **default** is "'CEE.SCEELKED"'.
PLILOAD	*dsname*	Specifies a data set that contains the PL/I compiler load module. *dsname* can include a member name. The **default** is "'IBM.SIBMZCMP(IBMZPLI)"'.
PLIPLNK	*dsname*	Specifies the data set that contains the IBM Environment prelink editor utility invocation load module that is to be used for preparing PL/I programs. *dsname* can include a member name. The **default** is "'CEE.SCEERUN(EDCPRLK)"'.
PLIPMSG	*dsname*	Specifies the data set that contains the message library that is to be used by the IBM prelink editor for preparing PL/I programs. *dsname* can include a member name. The **default** is "'CEE.SCEEMSGP(EDCPMSGE)"'.
POPTION	NONE *string*	Specifies a list of the C compiler language prelink utility options. For information on the options provided, refer to the *z/OS C/C++ User's Guide*. Use the **default,** NONE, to give no options.
PRECOMP	YES NO	Specifies whether to precompile.

Table 23. General DSNH CLIST parameters (continued)

Parameter	Value	Comments
PRELINK	YES NO	Specifies whether to execute the C compiler prelink utility to make your program reentrant. This utility concatenates compile-time initialization information (for writable static) from one or more text decks into a single initialization unit. If this step is requested, it must follow the compile step and precede the link-edit step. This parameter can apply to IBMCOB that also has a prelink step. Whether the prelink step applies to C or IBMCOB is determined by the choice of values C, CPP, or IBMCOB for the HOST parameter. Descriptions of the prelink process for C and IBMCOB are presented in their respective language publications. If PRELINK(YES) is specified or defaulted for a HOST language compiler that does not support the prelink utility, DB2 will issue warning message DSNH760I and prelink utility processing will be bypassed.
PRINT	NONE *dsname* LEAVE TERM	Specifies where to send printed output, including the lists of options, source, cross-reference, error, and summary information. The **default**, NONE, omits printed output. *dsname* specifies a data set that is to be used for the output. Do not enclose *dsname* between apostrophes. The current user profile is prefixed to *dsname*. The following suffixes are also added: • SYSCPRT.LIST for PL/I macro listings (these listings are overwritten by the compiler listings) • PCLIST for precompiler listings • CXLIST for CICS command translator listings • LIST for compiler listings The PRINT parameter is ignored for the compile step when HOST(CPP) is specified. • SYSOUT.PRELLIST for C prelink utility listings • LINKLIST for link-edit listings LEAVE sends output to the specified print data set. You can allocate the print data set in one of the following ways: • Dynamically • In the JCL that is used to run the DSNH CLIST (if in batch mode) • With the TSO ALLOCATE command (before running DSNH) TERM sends output to the terminal.
PSECSPAC	*integer*	Specifies the amount of secondary space to allocate for print data sets, in the units given by SPACEUN. The **default** is 20.
PSPACE	*integer*	Specifies the primary size of the print data sets in the units given by SPACEUN. The **default** is 20.
RCTERM	*integer*	Specifies the minimum value of the return code from the precompile step that prevents execution of later steps. The **default** is 8.

Table 23. General DSNH CLIST parameters (continued)

Parameter	Value	Comments
RUN	TSO or YES BATCH or NO CAF CICS IMS RRSAF	Specifies whether to execute the compiled program if the previous steps are successful, and, if so, in which environment it executes. Your choice for the RUN parameter might affect your choice for LLIB. TSO or YES indicates that the application program is to be scheduled for execution in the TSO environment, and executes the compiled program. BATCH or NO indicates that the application program is not to be scheduled for execution, and defaults to TSO as the execution environment. CAF indicates that the application program is to be scheduled for execution in the call attachment facility environment. Specify BATCH or NO with CAF to indicate that the application program is not to be scheduled for execution, but to identify CAF as the execution environment. (BATCH,CAF) or (NO,CAF) CICS indicates that the application program is not to be scheduled for execution, and identifies CICS as the run-time execution environment. CICS applications cannot run in TSO. IMS indicates that the application program is not to be scheduled for execution, and identifies IMS as the run-time execution environment. IMS applications cannot run in TSO. RRSAF indicates that the application program is not to be scheduled for execution, and identifies RRSAF as the run-time execution environment. RRSAF applications cannot run in TSO.
RUNIN	TERM *dsname* LEAVE NONE	Specifies where to get input for the RUN step. The **default**, TERM, gets input from the terminal. *dsname* specifies a data set that is to be used for the input. LEAVE gets input from SYSIN if the only steps taken are LINK and RUN. LEAVE gets input from FT05F001 if the language is Fortran. Do not use LEAVE in any other case. NONE allocates no input file.
RUNOUT	TERM *dsname* LEAVE NONE	Specifies where to send output from the RUN step. The **default**, TERM, sends output to the terminal. *dsname* specifies a data set to receive output. LEAVE sends output to SYSPRINT if the only steps taken are LINK and RUN. LEAVE sends output to FT06F001 if the language is Fortran. Do not use LEAVE in any other case. NONE allocates no output file for the RUN step.
SOMDLLI	*dsname*	Specifies the name that of the SOM/MVS DLL import library. This parameter is obsolete.
SOURCE	NO YES	Specifies whether the source code and diagnostics are to be printed with output from the precompiler, CICS command translator, and compiler.

Table 23. General DSNH CLIST parameters (continued)

Parameter	Value	Comments
SPACEUN	TRACK CYLINDER	Specifies the unit of space for PSPACE and WSPACE. TRACK makes the space unit one track. CYLINDER makes the space unit one cylinder.
SQL	DB2 ALL	Specifies how to interpret SQL statements and check syntax for use by either DB2 UDB for z/OS or other database management systems. The **default**, DB2, indicates that SQL statements are to be interpreted and syntax is to be checked for use by DB2 UDB for z/OS. SQL(DB2) is the recommended mode for DRDA access when the server is a DB2 subsystem. ALL indicates that SQL statements are to be interpreted for use by database management systems that are not DB2 UDB for z/OS. SQL syntax checking is deferred until bind time so that the remote location can bind the resulting DBRM. When SQL(ALL) is in effect, the precompiler issues an informational message if SAA® reserved words are used as identifiers. SQL(ALL) is the recommended mode if you have written your application to be executed in a environment that is not DB2 UDB for z/OS. The **default** is SQL(DB2).
SQLDELIM	DEFAULT APOSTSQL QUOTESQL	Specifies the APOSTSQL or QUOTESQL precompiler option, to set the SQL string delimiter and, by implication, the SQL escape character within SQL statements. Whichever character is chosen to be the string delimiter, the other is used for the SQL escape character. This parameter is effective only for COBOL. For PL/I, Fortran, and assembler language programs, the precompiler forces the APOSTSQL option. DEFAULT designates the value that is chosen during installation for the SQL STRING DELIMITER field on the APPLICATION PROGRAMMING DEFAULTS panel. APOSTSQL specifies that the string delimiter is the apostrophe (') and the escape character is the quotation mark ("). QUOTESQL specifies that the string delimiter is the quotation mark (") and the escape character is the apostrophe (').
SQLFLAG	IBM or SAA STD or 86 *ssname* *qualifier*	This option is no longer supported.
STDSQL	NO YES or 86	Specify whether to interpret SQL using a subset of ANSI rules. NO specifies that DB2 rules are used. YES or 86 automatically implies that the NOFOR option is used.
SUFFIX	YES NO	Specifies whether the TSO standard naming convention must be followed. That convention adds a TSO authorization ID prefix and a host language suffix to the name of the input data set (unless that name is enclosed between apostrophes, or already ends in the appropriate suffix). For example, names become *userid.name*.COBOL, *userid.name*.PLI, *userid.name*.Fortran, or *userid.name*.ASM.

Table 23. General DSNH CLIST parameters (continued)

Parameter	Value	Comments
SYSTEM	*subsystem-name*	Specifies the DB2 subsystem name as it is known to the z/OS operating system. The **default** is the installation-defined subsystem name (often DSN).
TERM	TERM *dsname* LEAVE NONE	Specifies where to send terminal output, including error information, error statements, and summary information. The **default**, TERM, sends output to the terminal. *dsname* specifies a data set that is to be used for terminal output. Do not enclose *dsname* between apostrophes. The following suffixes are added to *dsname*: • PCTERM for precompiler output • LIST for compiler output LEAVE sends the output to the current allocation for SYSTERM. NONE omits terminal output.
TIME	ISO JIS USA EUR LOCAL	Specifies the format for time values that are to be returned, overriding the format that is specified as the location default. There is **no default** because this option overrides the default previously specified.
VERSION	*version-id* AUTO	Specifies the name of the version ID for the program and associated DBRM during the DB2 precompile step. AUTO specifies that the consistency token is used to generate the version ID. If the consistency token is a timestamp, the timestamp is converted into ISO character format and used as the version identifier. The **default** is no version ID if specified at precompiler invocation.
WORKUNIT	*unit*	Specifies the device to use for print and work data sets. *unit* can be a unit name or a device type. The **default** in batch mode is any eligible device. The **default** in any other mode is the UADS unit name for the current TSO user.
WSECSPAC	*integer*	Specifies the amount of secondary space to allocate for work data sets, in the units given by SPACEUN. The **default** is 20.
WSPACE	*integer*	Specifies the primary size of the work data sets in the units given by SPACEUN. The **default** is 20.
XLIB	*dsname*	Specifies the linkage editor include library that is to be used for DB2 routines. The **default** is '"*prefix*.SDSNLOAD"'.
XREF	NO YES	Specifies whether a sorted cross-reference listing of symbolic names that are used in source statements is to be printed with output from the precompiler.

Note: Precompiler options do not affect ODBC behavior.

DSNH/DSN subcommand summary

Table 24 and Table 25 on page 257 differentiate the functions that support BIND PLAN and BIND PACKAGE. Each table associates the DSNH CLIST parameter and its corresponding DSN BIND PLAN or BIND PACKAGE subcommand keyword, if any. In general:

- The function and value of a CLIST parameter is identical to that of its corresponding DSN subcommand keyword unless otherwise noted.
- A DSNH parameter value of NONE indicates that the corresponding DSN keyword is not specified on subcommand invocation. Exceptions are noted where applicable.

DSNH CLIST/BIND PLAN subcommand comparison

Table 24. DSNH CLIST/ BIND PLAN subcommand summary

DSNH CLIST		BIND PLAN subcommand		
Parameter	**Value**	**Keyword**	**Value**	**Comments**
ACQUIRE	USE ALLOCATE	ACQUIRE	USE ALLOCATE	
ACTION	REPLACE ADD	ACTION	REPLACE ADD	
BDMEM	DEFAULT[1] *dbrm-member-name* NONE[2]	MEMBER	*dbrm-member-name*	[1] DBRM member name, which is obtained from one of the following sources, in the order listed: • BDBRMLIB member name • DBRMLIB member name • INPUT member name, or generated using *dsname*. [2] Keyword is not specified on subcommand invocation.
BIND	YES[1] NO[2]	*(command-verb)*		[1] Execute BIND PLAN subcommand. [2] Do not execute BIND PLAN subcommand.
BLIB	NONE[1] *dsname*	LIBRARY	*dbrm-pds-name*	[1] Keyword is not specified on subcommand invocation.
B*n*LIB[1]	NONE[2] *dsname*	LIBRARY	*list of dbrm-pds-names*	[1] *n* can be 2, 3, 4, 5, 6, 7, or 8. Specify the first data set name by using the BLIB parameter. Specify any additional data set names by using this parameter. [2] No additional data set names.

Table 24. DSNH CLIST/ BIND PLAN subcommand summary (continued)

DSNH CLIST		BIND PLAN subcommand		
Parameter	**Value**	**Keyword**	**Value**	**Comments**
BMEM[1]	NONE[2] *list of dbrm-member -names*	MEMBER	*list of dbrm-member-names*	[1] Specify the first DBRM member name using the BDMEM parameter and any additional member names individually using this parameter. [2] No additional DBRM member names.
CACHESIZE	NONE[1] *decimal-value*[2]	CACHESIZE	*decimal-value*[2]	[1] The size is provided by the subsystem. [2] Specify a size from 0 to 4096 bytes.
CICS	NONE[1] *application-ids*	CICS	*application-ids*	[1] Keyword is not specified on subcommand invocation.
CURRENTDATA	YES NO NONE	CURRENTDATA	YES NO	
CURRENTSERVER	NONE[1] *location-name*	CURRENTSERVER	*location-name*	
DBPROTOCOL	NONE DRDA PRIVATE	DBPROTOCOL	DRDA PRIVATE	If you specify PRIVATE, your application cannot include SQL statements that were added to DB2 after Version 7.
BDBRMLIB	DEFAULT[1] *dsname(member)* NONE[2]	LIBRARY	*dbrm-pds-name*	[1] The precompiler DBRMLIB data set is used. If the precompiler DBRMLIB is not specified, the default-generated DBRMLIB library that is based on the INPUT data set is used. [2]Keyword is not specified on subcommand invocation.
DEFER	NONE[1] PREPARE	DEFER	PREPARE	[1] Keyword is not specified on subcommand invocation.
DEGREE	1 ANY	DEGREE	1 ANY	
DISABLE	NONE BATCH CICS DB2CALL IMS DLIBATCH IMSBMP IMSMPP RRSAF	DISABLE	NONE BATCH CICS DB2CALL IMS DLIBATCH IMSBMP IMSMPP RRSAF	
DISCONNECT	EXPLICIT AUTOMATIC CONDITIONAL	DISCONNECT	EXPLICIT AUTOMATIC CONDITIONAL	

Table 24. DSNH CLIST/ BIND PLAN subcommand summary (continued)

DSNH CLIST		BIND PLAN subcommand		
Parameter	**Value**	**Keyword**	**Value**	**Comments**
DLIBATCH	NONE[1] *list of connection-ids*	DLIBATCH	*connection-name*	[1]Keyword is not specified on subcommand invocation.
DYNAMICRULES	RUN BIND	DYNAMICRULES	RUN BIND	
ENABLE	NONE * BATCH CICS DB2CALL IMS DLIBATCH IMSBMP IMSMPP RRSAF	ENABLE	NONE * BATCH CICS DB2CALL IMS DLIBATCH IMSBMP IMSMPP RRSAF	
EXPLAIN	NO YES	EXPLAIN	NO YES	
FLAG	I C E W	FLAG	I C E W	
IMSBMP	NONE[1] *imsid*	IMSBMP	*imsid*	[1]Keyword is not specified on subcommand invocation.
IMSMPP	NONE[1] *imsid*	IMSMPP	*imsid*	[1]Keyword is not specified on subcommand invocation.
ISOLATION	RR RS CS UR	ISOLATION	RR RS CS UR	
KEEPDYNAMIC	NO YES	KEEPDYNAMIC	NO YES	
NODEFER	NONE[1] PREPARE	NODEFER	PREPARE	[1]Keyword is not specified on subcommand invocation.
OPTHINT	**(' ')** *('hint-id')*	OPTHINT	**(' ')** *('hint-id')*	
OWNER	NONE[1] *authorization-id*	OWNER	*authorization-id*	[1]Keyword is not specified on subcommand invocation.
PATH	*(schema-name)* (**USER**)*(schema-name,* **USER***...)*	PATH	*(schema-name)* (**USER**)*(schema-name,* **USER***...)*	
PKLIST	NONE[1] *list of collection-ids and package-names*	PKLIST	*list of collection-ids and package- names*	[1] The package names are not specified on subcommand invocation.

Table 24. DSNH CLIST/ BIND PLAN subcommand summary (continued)

DSNH CLIST		BIND PLAN subcommand		
Parameter	**Value**	**Keyword**	**Value**	**Comments**
PLAN	*plan-name*[1]	PLAN *(primary-keyword)*	*plan-name*	[1] *plan-name* must not be DEFAULT. The default *plan-name* is the first of the following available choices that are defined in the INPUT data set: • DBRM member name • Left-most qualifier If no name is found, a plan is not created.
QUALIFIER	NONE[1] *implicit-qualifier*	QUALIFIER	*qualifier-name*	[1]Keyword is not specified on subcommand invocation.
RELEASE	COMMIT DEALLOCATE	RELEASE	COMMIT DEALLOCATE	
REOPT	NONE[1] VARS	REOPT	NONE ALWAYS ONCE	[1]Keyword is not specified on subcommand invocation.
RETAIN	NO[1] YES[2]	RETAIN		[1]Keyword is not specified on subcommand invocation. [2] Keyword is specified on subcommand invocation.
SQLRULES	DB2 STD	SQLRULES	DB2 STD	
VALIDATE	RUN BIND	VALIDATE	RUN BIND	

DSNH CLIST/BIND PACKAGE subcommand comparison

Table 25. DSNH CLIST/ BIND PACKAGE subcommand summary

DSNH CLIST		BIND PACKAGE subcommand		
Parameter	**Value**	**Keyword**	**Value**	**Comments**
PACTION	REPLACE ADD	ACTION	REPLACE ADD	
PBIND	NO[1] YES[2]	*(command-verb)*		[1] Do not execute BIND PACKAGE subcommand. [2] Execute BIND PACKAGE subcommand.
PCICS	NONE[1] *application-ids*	CICS	*application-ids*	[1]Keyword is not specified on subcommand invocation.
COPY	NONE[1] *collection-id.* *package-id*	COPY	*collection-id.* *package-id*	[1]Keyword is not specified on subcommand invocation.
COPYVER	*version-id*	COPYVER	*version-id*	
PCURRENTDATA	NO YES NONE	CURRENTDATA	YES NO	

Table 25. DSNH CLIST/ BIND PACKAGE subcommand summary (continued)

DSNH CLIST		BIND PACKAGE subcommand		
Parameter	**Value**	**Keyword**	**Value**	**Comments**
PDBPROTOCO	NONE DRDA PRIVATE	DBPROTOCOL	DRDA PRIVATE	If you specift PRIVATE, your application cannot include SQL statements that were added to DB2 after Version 7.
PDBRMLIB	DEFAULT[1] *dsname(member)* NONE[2]	LIBRARY	*dbrm-pds-name*	[1] The precompiler DBRMLIB data set is used. If the precompiler DBRMLIB is not specified, the default-generated DBRMLIB library that is based on the INPUT data set is used. [2]Keyword is not specified on subcommand invocation.
PDEFER	NONE[1] PREPARE	DEFER	PREPARE	[1]Keyword is not specified on subcommand invocation.
PDEGREE	1 ANY	DEGREE	1 ANY	
PDISABLE	NONE BATCH CICS DB2CALL IMS DLIBATCH IMSBMP IMSMPP REMOTE RRSAF	DISABLE	NONE BATCH CICS DB2CALL IMS DLIBATCH IMSBMP IMSMPP REMOTE RRSAF	
PDLIBATCH	NONE[1] *list of connection-ids*	DLIBATCH	*connection-name*	[1]Keyword is not specified on subcommand invocation.
PDMEM	DEFAULT[1] *dbrm-member-name* NONE[2]	MEMBER	*dbrm-member- name*	[1] DBRM member name, which is obtained from one of the following sources, in the order listed: • PDBRMLIB member name • DBRMLIB member name • INPUT member name, or generated using *dsname* [2]Keyword is not specified on subcommand invocation.
PDYNAMICRULES	NONE RUN BIND DEFINE INVOKE	DYNAMICRULES	RUN BIND DEFINE INVOKE	

Table 25. DSNH CLIST/ BIND PACKAGE subcommand summary (continued)

DSNH CLIST		BIND PACKAGE subcommand		
Parameter	**Value**	**Keyword**	**Value**	**Comments**
PENABLE	NONE * BATCH CICS DB2CALL IMS DLIBATCH IMSBMP IMSMPP REMOTE RRSAF	ENABLE	NONE * BATCH CICS DB2CALL IMS DLIBATCH IMSBMP IMSMPP REMOTE RRSAF	
EXPLAIN	NO YES	EXPLAIN	NO YES	
PFLAG	I C E W	FLAG	I C E W	
PIMSBMP	NONE[1] *imsid*	IMSBMP	*imsid*	[1]Keyword is not specified on subcommand invocation.
PIMSMPP	NONE[1] *imsid*	IMSMPP	*imsid*	[1]Keyword is not specified on subcommand invocation.
PISOLATION	NONE[1] RR RS CS UR NC	ISOLATION[1]	RR RS CS UR NC	[1] For local packages, the default value is the same as that of the plan that is appended at execution time. For remote packages, the default value is RR.
PKEEPDYNAMIC	NONE NO YES	KEEPDYNAMIC	NO YES	
PNODEFER	NONE[1] PREPARE	NODEFER	PREPARE	[1]Keyword is not specified on subcommand invocation.
POPTHINT	**(' ')** (*'hint-id'*)	OPTHINT	**(' ')** (*'hint-id'*)	
POWNER	NONE[1] *authorization-id*	OWNER	*authorization-id*	[1]Keyword is not specified on subcommand invocation.
PACKAGE	DEFAULT[1] *location-name. collection-id*	PACKAGE	*location-name. collection-id*	[1] Member name that is defined in the INPUT parameter data set, or the data set name if no member name was specified.
PPATH	*(schema-name)* (**USER**)*(schema-name*, **USER**, *...)*	PATH	*(schema-name)* (**USER**)*(schema-name*, **USER**, *...)*	
PQUALIFIER	NONE[1] *implicit-qualifier*	QUALIFIER	*qualifier-name*	[1]Keyword is not specified on subcommand invocation.

Table 25. DSNH CLIST/ BIND PACKAGE subcommand summary (continued)

DSNH CLIST		BIND PACKAGE subcommand		
Parameter	**Value**	**Keyword**	**Value**	**Comments**
PRELEASE	NONE[1] COMMIT DEALLOCATE	RELEASE[1]	COMMIT DEALLOCATE	[1] For local packages, the default value is the same as that of the plan that is appended at execution time. For remote packages, the default value is NONE.
REOPT	NONE[1] VARS	REOPT	NONE ALWAYS ONCE	[1]Keyword is not specified on subcommand invocation.
REMOTE	NONE[1] *location-name,* *<luname>*	REMOTE	*network-name*	[1]Keyword is not specified on subcommand invocation.
REPLVER	NONE[1] *version-id*	REPLVER	*version-id*	[1] *version-id* is not specified on subcommand invocation.
SQLERROR	NOPACKAGE CONTINUE	SQLERROR	NOPACKAGE CONTINUE	
PVALIDATE	RUN BIND	VALIDATE	RUN BIND	

Usage notes

CICS translator: Do not use CICS translator options in the source language for assembler programs; pass the options to the translator with the CICSOPT option.

COBOL options: The COBOL DYNAM option has several restrictions:

- You cannot use the option with CICS.
- You must use the VS COBOL II library or the Language Environment (z/OS Language Environment) library.
- To use the option with TSO or batch, the SDSNLOAD library must precede the IMS RESLIB in the step library, job library, or link list concatenations.
- To use the option with IMS, the IMS RESLIB must precede DSNLOAD.

Several COBOL options require DD statements that are not provided by the DSNH CLIST, as shown in Table 26.

Table 26. COBOL options that require additional DD statements

Option	Statements required for...
CDECK	SYSPUNCH
COUNT	SYSCOUNT, SYSDBG, SYSDBOUT, SYSUT5, a debug file
DECK	SYSPUNCH
DUMP	SYSABEND, SYSDUMP, or SYSUDUMP
FDECK	SYSPUNCH
FLOW	SYSCOUNT, SYSDBG, SYSDBOUT, SYSUT5, a debug file
LVL	SYSUT6
STATE	SYSCOUNT, SYSDBG, SYSDBOUT, SYSUT5, a debug file
SYMDUMP	SYSCOUNT, SYSDBG, SYSDBOUT, SYSUT5, a debug file

Table 26. COBOL options that require additional DD statements (continued)

Option	Statements required for...
SYST	SYSOUT
SYSx	SYSOUx
TEST	SYSUT5

COBOL parameters: The BUF and SIZE parameters passed to the COBOL compiler might need to be changed.

COPTION: Do not use the COPTION parameter to specify values for the LINECOUNT, SOURCE, TERM, and XREF compiler options; use the DSNH LINECOUNT, SOURCE, TERM, and XREF keywords.

Fortran and PL/I considerations: Variable-format input records are not supported.

Library limits: At most, eight bind libraries, four precompile libraries, four compile libraries, and four link-edit libraries can exist.

User-supplied DSNHDECP module: The following steps are required to enable DSNH CLIST to load your user-supplied DSNHDECP module rather than the DB2-supplied DSNHDECP module:

1. The JOBLIB or STEPLIB concatenation of any job or TSO userid that calls DSNH must allocate the library where the user-supplied DSNHDECP module resides (usually *prefix*.SDSNEXIT) before it allocates the library where the DB2-supplied DSNHDECP module resides (*prefix*.SDSNLOAD).
2. The DSNH call should include the PCLOAD parameter, coded as follows:
   ```
   PCLOAD('*(DSNHPC)')
   ```

Link-edit:

- DSNH cannot process programs that need additional link-edit control statements and cannot link-edit programs that use the call attachment facility.
- You cannot use the NOLOAD and SYNTAX link-edit options.

NONE is a reserved word: NONE cannot be the name of an input or a load library, or the value of the string passed with PARMS.

SQL host variables: You must explicitly define SQL host variables.

SYSPROC: If compilation is done, the SYSPROC data set must include the DB2 CLIST library.

WORKUNIT parameter: You must use the WORKUNIT parameter when running the DSNH CLIST in batch mode. This insures that the temporary and intermediate data sets are allocated to the correct devices.

Examples

Example 1: Precompile, bind, compile, link-edit, and run the COBOL program in data set *prefix*.SDSNSAMP(DSN8BC4).

- The compiler load module is in SYS1.LINKLIB (IKFCBL00).
- Additional load modules to be included are in *prefix*.RUNLIB.LOAD and *prefix*.SDSNSAMP.

- The load module is to be put into the data set *prefix*.RUNLIB.LOAD(DSN8BC4).
- The plan name is DSN8BC81 for the bind and run.
- DCLGEN data from *prefix*.SRCLIB.DATA is required for the precompile.

This example assumes that the DSNH CLIST is in your SYSPROC concatenation.

```
DSNH INPUT('prefix.SDSNSAMP(DSN8BC4)'') -
  COBLOAD('SYS1.LINKLIB(IKFCBL00)'') -
  LLIB('prefix.RUNLIB.LOAD'') -
  L2LIB('prefix.SDSNSAMP'') -
  LOAD('prefix.RUNLIB.LOAD'') -
  PLAN(DSN8BC81) -
  PLIB('prefix.SRCLIB.DATA'')
```

Example 2: Precompile, bind, compile, and link-edit the program in data set *prefix*.SDSNSAMP.PLI(DSN8BP4).

- The program is written in PL/I; the macro pass is not needed.
- The PL/I compiler options MAP and LIST are to be used.
- Additional load modules to be included are in *prefix*.RUNLIB.LOAD and *prefix*.SDSNSAMP.
- The PL/I optimizing compiler load module is in library SYS2.LINKLIB(IEL0AA).
- The DB2 subsystem identifier is SSTR.
- The load module is put into the data set *prefix*.RUNLIB.LOAD(DSN8BC4).
- Printed output is sent to the following data sets:

Output	Data set
Precompiler listings	*prefix*.PROG.PCLIST
Compiler listings	*prefix*.PROG.LIST
Link-edit listings	*prefix*.PROG.LINKLIST

- The plan name is DSN8BC81 for the bind and run.
- The DCLGEN data from *prefix*.SRCLIB.DATA is required for the precompile.

```
DSNH INPUT('prefix.SDSNSAMP(DSN8BP4)'') -
  HOST(PLI) MACRO(NO) -
  COPTION ('MAP LIST') -
  LLIB('prefix.RUNLIB.LOAD'') -
  L2LIB('prefix.SDSNSAMP'') -
  PLILOAD('SYS2.LINKLIB(IEL0AA)'') -
  SYSTEM(SSTR) -
  LOAD('prefix.RUNLIB.LOAD'') -
  PRINT(PROG) -
  PLAN(DSN8BC81) -
  PLIB('prefix.SRCLIB.DATA'')
```

The COPTION parameters are enclosed between single apostrophes so that they are passed by TSO as a single parameter. If a single token is being passed as a parameter, no apostrophes are needed. That same rule applies to the PARMS and CICSOPT parameters.

If a data set name is being passed as a parameter, and you want TSO to add your user prefix, no apostrophes are needed. If the usual TSO prefixing and suffixing must not be performed, the data set name must be enclosed between sets of three apostrophes if the CLIST is executed implicitly, and sets of six apostrophes if the CLIST is executed explicitly.

The user prefix for that example is *prefix*; if it had been SMITH, the listing data set names would be as shown in the preceding example, except that SMITH would be used as the first level qualifier. For example, the compiler listings would have gone to SMITH.PROG.LIST.

Example 3: Invocation of the DB2-C sample application program *prefix*.SDSNSAMP(DSN8BD3).

- The C linkage editor include library is EDC.V1R1M1.SEDCBASE
- The C compiler load module is EDC.V1R1M1.SEDCCOMP(EDCCOMP)
- Printed output is sent to the following data sets:

Output	Data set
Precompiler listings	*user_id*.TEMP.PCLIST
Compiler listings	*user_id*.TEMP.SYSCPRT.LIST
Prelink utility listings	*user_id*.TEMP.SYSOUT.PRELLIST
Link-edit listings	*user_id*.TEMP.LINKLIST

- The following C DD names are allocated based on the PRINT keyword value:

DD name	Allocation
SYSCPRT	Used in the compile step
SYSUT10	Used in the compile step
SYSOUT	Used in the prelink step

  SYSUT10 and SYSCPRT are always allocated to the same data set or destination.
- SYSTERM is used in the compile step. It is based on the TERM keyword.
- CEEDUMP is used in the run step. It is based on the RUNOUT keyword.
- The LOPTION keyword values of AMODE(31) and RMODE(ANY) are required when link editing the C sample program to insure 31-bit addressability during execution.

```
ALLOC      DD(SYSPROC) DSN('prefix.SDSNCLST ') SHR
%DSNH BIND(YES) ACQUIRE(USE) ACTION(REPLACE)-
 EXPLAIN(NO) -
 CICSXLAT(NO) -
 COMPILE(YES) -
 CCLLIB('EDC.V1R1M1.SEDCBASE'')-
 CCLOAD('EDC.V1R1M1.SEDCCOMP(EDCCOMP)'')-
 DBRM('prefix.DBRMLIB.DATA(DSN8BD3)'')-
 DECIMAL(PERIOD) DELIMIT(DEFAULT) FLAG(I)-
 HOST(C) ISOLATION(RR)-
 INPUT('prefix.SDSNSAMP(DSN8BD3)'')-
 LINK(YES)-
 LLIB('prefix.RUNLIB.LOAD'')-
 L2LIB('prefix.SDSNLOAD'')-
 LOAD('prefix.RUNLIB.LOAD'')-
 LOPTION('AMODE(31) RMODE(ANY)')-
 MACRO(NO)-
 OUTNAME(TEMP)-
 PLAN(DSN8BD31) PRECOMP(YES)-
 PLIB('prefix.SDSNSAMP'')-
 PRELINK(NO)-
 POPTION(NONE)-
 PRINT(TEMP) RCTERM(8)-
 RELEASE(COMMIT) RETAIN(YES)-
 RUN(NO) RUNIN(TERM)-
 RUNOUT(TERM) SOURCE(YES)-
 SYSTEM(DSN) SQLDELIM(DEFAULT)-
 VALIDATE(RUN)
```

Chapter 42. END (DSN)

The DSN subcommand END is used to end the DSN session and return to TSO.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Usage note"
- "Example"

Environment

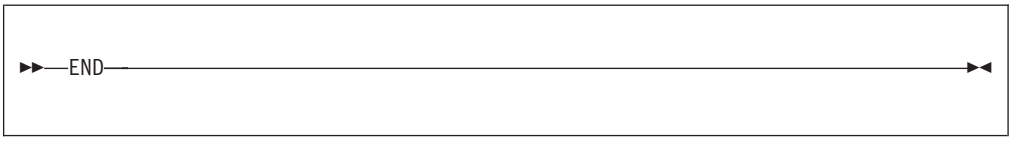
This subcommand originates from a TSO input stream when DSN is running in either background or foreground mode.

Data sharing scope: Member

Authorization

None is required.

Syntax



Usage note

Ending the DSN session in batch or foreground: In batch, if END is not found in the SYSIN stream, /* or // ends the DSN session. From the foreground, pressing the ATTENTION key twice ends the DSN session.

Example

End the DSN session and return to TSO.

```
TSO prompt :  READY
USER enters:  DSN SYS (SSTR)
DSN prompt :  DSN
USER enters:  RUN PROGRAM (MYPROG)
DSN prompt :  DSN
USER enters:  END
TSO prompt :  READY
```

© Copyright IBM Corp. 1983, 2008

Chapter 43. FREE PACKAGE (DSN)

The DSN subcommand FREE PACKAGE can be used to delete a specific version of a package, all versions of a package, or whole collections of packages.

The FREE PACKAGE subcommand deletes corresponding table entries from the catalog tables. Authorization for a package name is removed only when no more versions of the package exist. After a version of a package has been freed, that package name is then available for use in a BIND PACKAGE subcommand to create a new package.

The FREE PACKAGE subcommand does not proceed until all currently running applications using the package finish running.

For additional information about packages, see Part 5 of *DB2 Application Programming and SQL Guide*.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 268
- "Option descriptions" on page 268
- "Usage notes" on page 269
- "Examples" on page 269

Environment

You can enter this subcommand from DB2I, or from a DSN session under TSO that is running in either foreground or background.

Data sharing scope: Group

Authorization

To execute this subcommand, you must use a privilege set of the process that includes one of the following privileges or authorities:

- Ownership of the package
- BINDAGENT privilege granted by the owner of the package
- SYSCTRL authority
- SYSADM authority
- PACKADM authority for the collection or for all collections

The BIND privilege on a package is *not* sufficient to allow a user to free a package.

© Copyright IBM Corp. 1983, 2008

Syntax

Option descriptions

location-name
Specifies the location of the DBMS where the package is to be freed. The location name must be defined in the SYSIBM.LOCATIONS table. If this table does not exist or the DBMS is not found, you receive an error message. If the location name is specified, the name of the local DB2 subsystem must be defined. See Part 3 of *DB2 Installation Guide* for information on how to define a location name within SYSIBM.LOCATIONS.

The **default** is the local DB2 subsystem if you omit *location-name*.

collection-id **or (*)**
Identifies the collection of the package to be freed. There is no default.

You can use an asterisk (*) to free all local packages with the specified *package-id* in all the collections that you are authorized to free. (You cannot use the * to free remote packages.)

package-id **or (*)**
Identifies the package to be freed. There is no default.

You can use an asterisk (*) to free all local packages in *collection-id* that you are authorized to free. (You cannot use the * to free remote packages.)

version-id **or (*)**
Identifies the version of the package to be freed.

You can use an asterisk (*) to free all local packages in the *collection-id* and *package-id* that you are authorized to free. (You cannot use the * to free remote packages.)

If you specify () for *version-id*, the empty string is used for the version ID.

If you omit the *version-id*, the default depends on how you specify *package-id*. If you use * for *package-id*, *version-id* defaults to *. If you provide an explicit value for *package-id*, *version-id* defaults to an empty string.

DBRMs created before DB2 Version 2 Release 3 use an empty string for *version-id* by default.

(*)
Frees all local DB2 packages that you are authorized to free.

Specifying (*) is equivalent to specifying the package name as (*.*.(*)) or (*.*).

FLAG
Indicates what messages you want the system to display. Use one of the following values to show messages of the corresponding types.

(I) All: informational, warning, error, and completion messages.

(W) Only warning, error, and completion messages.

(E) Only error and completion messages.

(C) Only completion messages.

Usage notes

Freeing multiple packages: If you free multiple packages with this subcommand, each successful free is committed before freeing the next package.

If an error occurs on a certain package specified explicitly in a list or implicitly with (*), FREE PACKAGE terminates for that package and continues with the next package to be processed.

Freeing trigger packages: You cannot free a trigger package using the FREE PACKAGE subcommand.

For more information about dropping triggers, see Chapter 5 of *DB2 SQL Reference*.

Examples

Example 1: Free version *newver* of the package TEST.DSN8BC81 located at USIBMSTODB22. Generate only warning, error, and completion messages (not informational messages).

```
FREE PACKAGE (USIBMSTODB22.TEST.DSN8BC81.(newver)) FLAG(W)
```

Example 2: Free all packages at the local server in the collection named TESTCOLLECTION.

```
FREE PACKAGE (TESTCOLLECTION.*)
```

Chapter 44. FREE PLAN (DSN)

The DSN subcommand FREE PLAN deletes application plans from DB2.

The FREE PLAN subcommand deletes corresponding table entries from the SYSIBM.SYSPLAN catalog tables. All authorization against an application plan name is dropped. The application plan name is then available for use in a BIND PLAN subcommand to create a new package.

The FREE PLAN subcommand does not proceed until all currently executing applications using that plan finish executing.

For additional information on plans, see Part 5 of *DB2 Application Programming and SQL Guide*.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 272
- "Option descriptions" on page 272
- "Usage notes" on page 272
- "Example" on page 272

Environment

You can enter this subcommand from DB2I, or from a DSN session under TSO that is running in either foreground or background.

Data sharing scope: Group

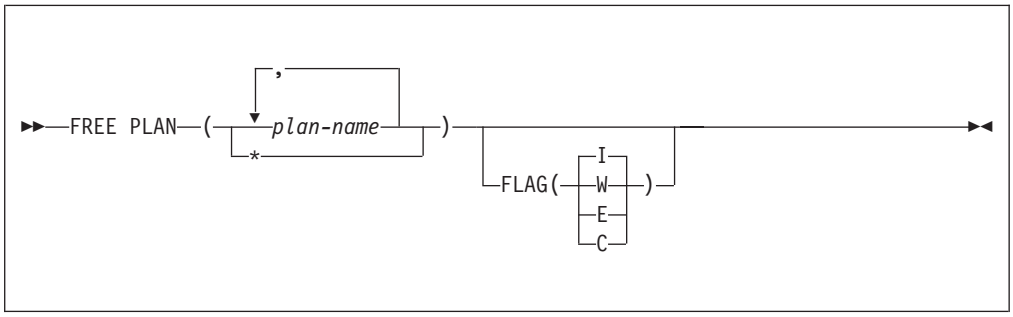
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- Ownership of the plan
- BIND privilege on the plan
- BINDAGENT privilege granted by the plan owner
- SYSCTRL authority
- SYSADM authority

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

(*plan-name***, ...)**
Lists the names of one or more plans you want to free.

(*) Frees *all* application plans over which you have BIND authority. Be careful when using this form of the command.

FLAG
Indicates what messages you want the system to display. Use one of the values listed to show messages of the corresponding types.

(I) All: informational, warning, error, and completion messages.

(W) Only warning, error, and completion messages.

(E) Only error and completion messages.

(C) Only completion messages.

Usage notes

Freeing multiple plans: If you free multiple plans with this subcommand, each successful free is committed before freeing the next plan.

If an error occurs on a certain plan specified explicitly in a list or implicitly with (*), FREE PLAN terminates for that plan and continues with the next plan to be processed.

Example

Free plan DSN8BC81 from DB2. Generate only warning, error, and completion messages (not informational messages).

```
FREE PLAN (DSN8BC81) FLAG (W)
```

Chapter 45. MODIFY admtproc,APPL=SHUTDOWN (z/OS)

The MODIFY *admtproc*, APPL=SHUTDOWN command stops the administrative scheduler from accepting new requests and starting new task executions. It also shuts down the administrative scheduler.

When the SHUTDOWN option is specified, the administrative scheduler waits until the execution of all currently running tasks completes. When all running tasks are complete, the administrative scheduler terminates.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Example"

Environment

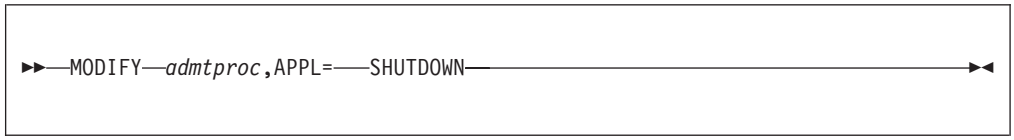
This command can be issued only from a z/OS console.

Data sharing scope: Member

Authorization

The command requires an appropriate level of operating system authority.

Syntax



You cannot include spaces when you specify options.

Option descriptions

Parameters must be separated by commas with no spaces.

admtproc
Specifies the procedure name of the administrative scheduler task that you want to modify.

Example

This command modifies the *admtproc* scheduler in order to shut it down.

Enter the following command on the system console:

```
modify admtproc,appl=SHUTDOWN
```

© Copyright IBM Corp. 1983, 2008

Chapter 46. MODIFY admtproc,APPL=TRACE (z/OS)

The MODIFY *admtproc*, APPL=TRACE command starts or stops traces in the administrative scheduler.

You do not need to stop the administrative scheduler to access the trace. To turn the trace on or off when the administrative scheduler starts, take one of the following actions:

- Modify the procedure parameter TRACE in the JCL job that starts the administrative scheduler, *admtproc*. This job is copied into one of the PROCLIB library during installation. Specify TRACE=ON or TRACE=OFF.
- Dynamically overwrite the trace parameter on the operator's console when you start the administrative scheduler. This operation can only be done manually, and cannot be performed when DB2 starts the administrative scheduler automatically.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Examples" on page 276

Environment

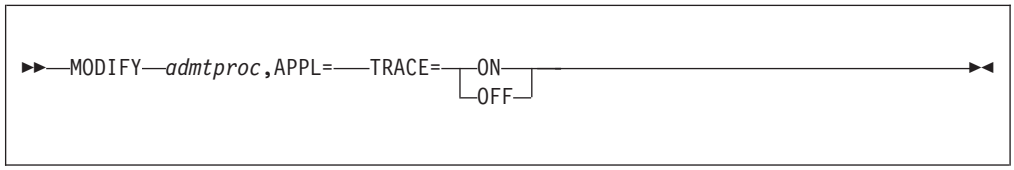
This command can be issued only from a z/OS console.

Data sharing scope: Member

Authorization

The command requires an appropriate level of operating system authority.

Syntax



You cannot include spaces when you specify options.

Option descriptions

admtproc
Specifies the procedure name of the administrative scheduler task that you want to modify.

© Copyright IBM Corp. 1983, 2008

ON
Turns the trace on.

OFF
Turns the trace off.

Examples

Example 1. This command modifies the *admtproc* scheduler, and turns on the trace.

Enter the following command on the system console:

```
modify admtproc,appl=trace=on
```

Response from z/OS console:

```
STC00072 DSNA672I MODIFY COMMAND FOR ADMIN SCHEDULER V91AADMT
NORMAL COMPLETION
```

Example 2. This command modifies the *admtproc* scheduler, and turns off the trace.

```
modify admtproc,appl=trace=off
```

Response from z/OS console:

```
STC00072 DSNA672I MODIFY COMMAND FOR ADMIN SCHEDULER V91AADMT
NORMAL COMPLETION
```

Chapter 47. MODIFY irlmproc,ABEND (z/OS IRLM)

The MODIFY *irlmproc*, ABEND command terminates IRLM abnormally. IRLM processes this command even if a DB2 subsystem is identified to it.

Abbreviation: F

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage notes" on page 278
- "Example" on page 278

Environment

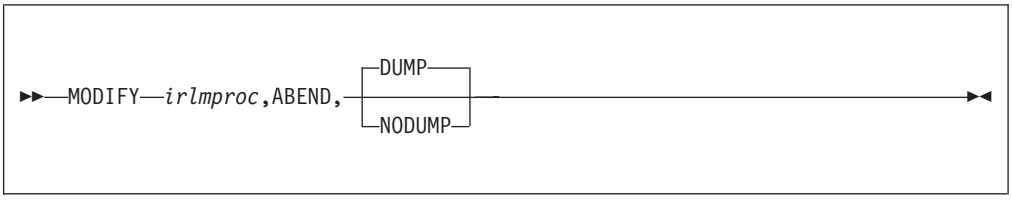
This command can be issued only from a z/OS console.

Data sharing scope: Member

Authorization

The command requires an appropriate level of operating system authority, as described in *z/OS MVS System Commands.*

Syntax



Option descriptions

Parameters must be separated by commas with no spaces.

irlmproc
Specifies the procedure name of the IRLM that is to be terminated.

DUMP
Specifies that IRLM is to terminate abnormally with a U2020 abend. A system dump is taken to the SYS1.DUMPxx data set. IRLM does not de-register from ARM.

NODUMP
Specifies that IRLM is to FORCE the DBMS off and terminate normally

© Copyright IBM Corp. 1983, 2008

without generating a dump. All DBMS work is quiesced and IRLM stops itself. NODUMP **requires** that IRLM be functioning normally. **Do not** use this option if IRLM appears to be hung.

A second invocation causes IRLM to terminate abnormally with a U2020 abend; no dump is taken.

Usage notes

Terminating IRLM: If any difficulties occur when terminating IRLM, see "Usage note" on page 413

Deregistering IRLM: You can use the NODUMP option to deregister IRLM before stopping it. This action prevents the automatic restart manager from immediately trying to restart IRLM.

Example

Enter the following command on the system console:

```
F KRLM001,ABEND
```

Response on the z/OS system console is as follows:

```
 DXR124E IR21001 ABENDED VIA MODIFY COMMAND
*IEA911E COMPLETE DUMP ON SYS1.DUMP00
  FOR ASID(0004)
   ERROR ID = SEQ00001 CPU00 ASID0004 TIME08.34.59.9
 DXR121I IR21001 END-OF-TASK CLEANUP SUCCESSFUL
 IEF450I IR21001 IR21001 - ABEND=S000 U2020 REASON=00000000
```

The default is dump. If you do not want a dump, you must specify the following command:

```
F KRLM001,ABEND,NODUMP
```

Chapter 48. MODIFY irlmproc,DIAG (z/OS IRLM)

The MODIFY *irlmproc*, DIAG command initiates diagnostic dumps for IRLM subsystems.

If IRLM detects a delay in the child-lock propagation process, it retries the XES calls in order to recover. Use the MODIFY *irlmproc*, DIAG command under the direction of IBM Software Support if this situation occurs.

Abbreviation: F

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage note" on page 280
- "Example" on page 280

Environment

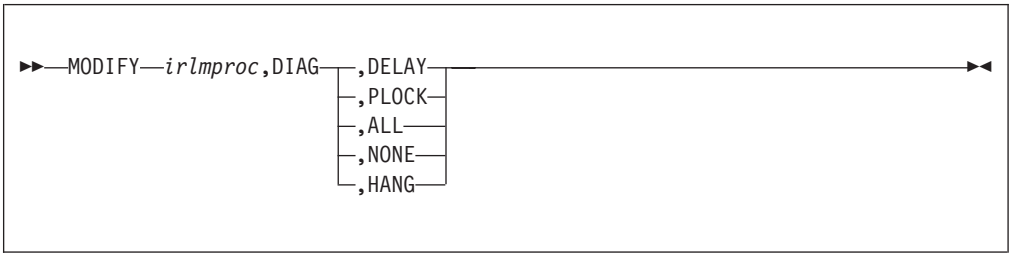
This command can be issued only from a z/OS console.

Data sharing scope: Group

Authorization

The command requires an appropriate level of operating system authority, as described in *z/OS MVS System Commands.*

Syntax



Option descriptions

Parameters must be separated by commas, with no spaces.

irlmproc
Specifies the procedure name of the IRLM instance that is to be diagnosed.

DIAG
Specifies that this is a diagnostic dump.

© Copyright IBM Corp. 1983, 2008

DELAY
Directs IRLM to generate a dump the first time it detects that child lock propagation to the coupling facility is taking longer than 45 seconds. The dump is placed in the SYS1.DUMP*xx* data set.

PLOCK
Directs IRLM to generate a dump the first time it detects that P-lock negotiation is taking longer than two minutes. Dumps of the IRLM and DB2 address spaces are placed in the SYS1.DUMPxx data set.

ALL
Directs IRLM to generate diagnostic dumps for IRLM or DBMS subsystems in a data sharing group for the following unusual conditions:

- P-lock negotiation takes longer than two minutes
- Child-lock propagation takes longer than 45 seconds
- If IRLM detects a delay in the child-lock propagation process, it retries the XES calls in order to recover.

NONE
Disables generating all diagnostic dumps.

HANG
Collects IRLM SYSPLEX dumps when DEADLOCK or TIMEOUT issues are suspected. The dumps are taken during DEADLOCK processing. The DEADLOCK processing is stopped, and the dynamic deadlock storage is collected. z/OS DUMP services then schedules an SRB to restart DEADLOCK processing. Message DXR183I is issued by each IRLM as DEADLOCK processing is restarted. If message DXR183I is not issued by an IRLM, that IRLM must be terminated and restarted. You must start the IRLM XCF CTRACE internally and wait 30 seconds before issuing this command.

Usage note

The MODIFY irlmproc,DIAG command should be used only under the direction of IBM Software Support.

This command is active for only one incident per IRLM, that is, after an IRLM instance detects the delay and initiates the dump. You can initiate one dump per IRLM in the group. You must re-enter the command to initiate another dump. Be aware that when you enter this command for one member of the data sharing group, **any** member that detects the delay initiates a dump.

The *irlmproc* identifies the procedure name for IRLM. If multiple IRLM instances exist in the same system, each procedure must have a unique name.

Example

Issue this command to initiate one diagnostic dump for the IR21PROC IRLM subsystem. The dump occurs once, after the propagation of child locks takes longer than 45 seconds.

```
MODIFY IR21PROC,DIAG,DELAY
```

Chapter 49. MODIFY irlmproc,PURGE (z/OS IRLM)

The MODIFY irlmproc,PURGE command releases IRLM locks retained due to a DB2, IRLM, or system failure. The command causes all retained locks for the specified DB2 to be deleted from the system, thereby making them available for update. Because retained locks protect updated resources, it should be used only after understanding what the resources are and the consequence to data integrity if they are deleted.

Abbreviation: F

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Syntax"
- "Usage notes" on page 282
- "Example" on page 282

Environment

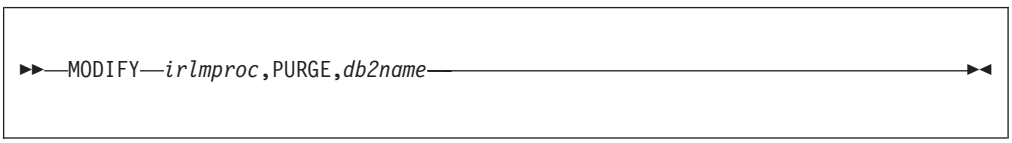
This command can be issued only from a z/OS console.

Data sharing scope: Member

Authorization

The command requires an appropriate level of operating system authority, as described in *z/OS MVS System Commands.*

Syntax



Option descriptions

Use commas with no spaces to separate parameters.

irlmproc
Specifies the active IRLM that is to process the command.

db2name
Specifies the inactive DB2 name, as displayed by the STATUS command.

© Copyright IBM Corp. 1983, 2008

Usage notes

DB2 subsystem inactive: The DB2 subsystem that owns the retained locks must be inactive or else this command fails.

The irlmproc must be the procedure name of an active IRLM that is connected to the same sysplex group as the failed member. Issuing a purge request using an inactive IRLM returns error IEE341I.

Example

Example:For an active DB2 subsystem named db2b with irlmproc name db2birlm, issue the following command to display all active and inactive subsystems in a data sharing sysplex:

```
F db2birlm,STATUS,ALLD
```

If the subsystem db2a is inactive, enter the following command:

```
F db2birlm,PURGE,db2a
```

Response on the MVS system console for completed purge request:

```
DXR109I IR2B002 PURGE COMMAND COMPLETED FOR DB2A
```

Explanation: In a sysplex environment, if the DB2 database is inactive and the database IRLM has stopped or is disconnected, the operator of the z/OS system uses one of the other active IRLM members to query retained locks and issue the PURGE request.

Chapter 50. MODIFY irlmproc,SET (z/OS IRLM)

The MODIFY irlmproc,SET command performs the following tasks:

- Dynamically sets the maximum private storage allowed from IRLM.
- Dynamically sets the number of trace buffers allowed for IRLM.
- Dynamically sets the number of LOCK LTE entries to be specified on the next connect to the XCF LOCK structure.
- Dynamically sets the timeout value for a specified subsystem.
- Dynamically sets the local deadlock frequency.

Abbreviation: F

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage notes" on page 285
- "Examples" on page 286

Environment

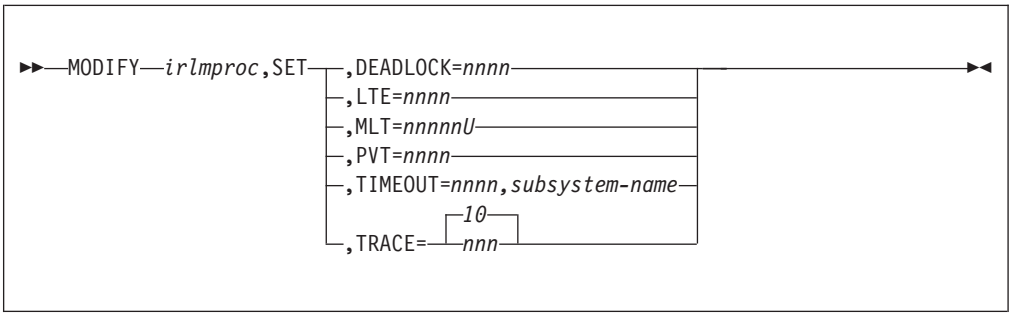
This command can be issued only from a z/OS console.

Data sharing scope: Group or Member, depending on whether you specify the DEADLOCK or LTE options.

Authorization

The command requires an appropriate level of operating system authority, as described in *z/OS MVS System Commands.*

Syntax



Option descriptions

Use commas with no spaces to separate parameters.

© Copyright IBM Corp. 1983, 2008

irlmproc
Specifies the IRLM that is to process the command.

SET
Sets the following values for this IRLM:

DEADLOCK=*nnnn*
Specifies the number, in milliseconds, indicating how often the local deadlock processing is scheduled. *nnnn* must be a number from 100 through 5000 milliseconds. If a member of a sysplex group and all IRLMs are not enabled for subsecond deadlock processing, message DXR106E is issued.

LTE=*nnnn*
Specifies the number of lock table entries that are to be specified on the next connect to the XCF lock structure. *nnnn* must be a number from 0 through 1024, and it must be an even power of 2. Each increment in value represents 1 048 576 LTE entries. Note that this parameter is used for data sharing only.

MLT=*nnnnU*
Specifies the upper limit of private storage above the two gigabyte bar, also called the MEMLIMIT, which is managed by MVS. This command allows you to dynamically change the MEMLIMIT for IRLM. This storage is used for locks. The value *nnnnn*. must be a five digit number from 0 through 99999, and *U* must be a one character units indicator with the value of M for megabytes, G for gigabytes, T for terabytes, or P for petabytes. MEMLIMIT specified for IRLM must be larger than both the 2 GB minimum and the amount of above the bar storage in use at the time of the command. If the specified value is out of range or lower than the amount in use (or the 2 GB minimum), then the command is rejected with message DXR106E. This MEMLIMIT update is temporary, holding only as long as the execution of this IRLM instance. To make a permanent MEMLIMIT change, update the corresponding IRLM startup procedure with a new value for the MEMLIMIT JCL EXEC parameter.

PVT=*nnnn*
Specifies the upper limit of private storage, below the two gigabyte bar. *nnnn* must be a four digit number from 1 through 1800. You can specify this value in megabytes or gigabytes by specifying M (for megabytes) or G (for gigabytes) after the value, as follows, *nnnn*M or *nnnn*G. IRLM monitors the amount of private storage used for locks. If the specified limit is reached, new lock requests will be rejected unless they are **must complete**. If the specified value is out of range or if IRLM's use of private storage is already larger than the specified value, the command is rejected with message DXR106E. No reserve for must complete locks is calculated from the specified PVT= value.

TIMEOUT=*nnnn,subsystem-name*
Requests that IRLM dynamically set the timeout value, in seconds, for the specified subsystem. *nnnn* must be a number from 1 through 3600. *subsystem-name* is the DB2 subsystem name, as displayed by the MODIFY irlmproc,STATUS command.

TRACE=*nnn*
Requests that IRLM dynamically set the maximum number of 64 KB trace buffers per trace type to the value you specify in *nnn*. *nnn* must

be a number from 10 through 255. If you specify a value outside of this range, IRLM automatically adjusts the value to a value within the range.

The **default** is **10**.

This value is used only when the external CTRACE writer is not active. The trace buffers are allocated from extended common storage area (ECSA).

IRLM does not immediately acquire the number of trace buffers you set using this command; IRLM allocates buffers as needed, not to exceed the number of buffers you specify. If the number of trace buffers that you set is less than the number of currently allocated buffers, IRLM brings the number within your specified range by releasing the oldest buffers at the end of the next deadlock cycle.

Usage notes

Effect of an IRLM restart: The values you set using the MODIFY irlmproc,SET command do not persist through a stop and restart of IRLM. The number of trace buffers for each trace type returns to the default value of 10.

TIMEOUT considerations: The TIMEOUT value must be a multiple of the local deadlock parameter. If the value entered is not an even multiple of the deadlock parameter, IRLM will increase the timeout value to the next highest multiple. This new value is used until the IRLM or identified subsystem is terminated, or the timeout is changed again by the operator. The value specified on the command does *not* affect the timeout value in the DB2 zparms.

Effect of the LTE parameter: If this IRLM is **not** connected to the group, but a valid value is specified, a message is issued stating that the value is set, but the value is not sent to any other member. If the member is already in the group, the value is sent to the Global Deadlock Manager (GDM) for IRLM, to be broadcast to all other members. If the GDM does not have the code applied, no response message is issued on any member. If the GDM has the code, all members with the applied code issue the response message as the command is processed. This value is used if the IRLM is the first to join the data sharing group, causing structure allocation, or the value is used during a REBUILD. If any IRLM joins later, it does not have the updated value. If multiple MODIFY commands are issued on the same or multiple IRLMs, some response messages might be missing. The last response message that is issued identifies the value to be used on the next CONNECT.

The number of lock table entries in the group is determined by the first IRLM to connect to the group during initial structure allocation or during REBUILD.

The LTE value is used in the following order:

1. The value specified using the MODIFY irlmproc,SET,LTE= command, if the value is greater than zero.
2. The value from the LTE= in the IRLMPROC, if the value is greater than zero.
3. The value determined by the existing logic, which divides the XES structure size returned on the XESQUERY call by 2 multiplied by the LTE width. The result is rounded to the nearest power of 2, which the existing logic uses for the value.

   Note: The LTE width is determined by the MAXUSRS value.

If an attempt is made to use a nonzero value from either option number 1 or 2, and that value is too large for the structure size that is returned on the QUERY, the value from the next option in the sequence is used instead.

Deadlock value range for non-supporting members: When an IRLM that supports subsecond deadlock joins a group that has a member that does not support subsecond deadlock, if the deadlock value of the new member is less than one second, the value is set to one second.

Examples

Example 1: Enter the following command on a z/OS system console:

```
F IR21PROC,SET,TRACE=20
```

Response on the z/OS system console is as follows:

```
DXR177I IR21033 THE VALUE FOR TRACE IS SET TO 20.
```

Example 2: Enter the following command on a z/OS system console:

```
F IR21PROC,SET,TIMEOUT=60,DBMS
```

Response on the z/OS system console is as follows:

```
DXR177I IR21033 THE VALUE FOR TIMEOUT IS SET TO 60 FOR DBMS
```

Example 3: Enter the following command on a z/OS system console:

```
F IR21PROC,SET,LTE=1024
```

Response on the z/OS system console is as follows:

```
DXR177I IR21033 THE VALUE FOR LTE IS SET TO 1024
```

Example 4: Enter the following command on a z/OS system console:

```
F IR21I,SET,DEADLOCK=1000
```

Response on the z/OS system console is as follows:

```
DXR177I IR21033 THE VALUE FOR DEADLOCK IS SET TO 1000 MILLISECONDS
```

Example 5: Enter the following command on a z/OS console:

```
F IR21I,SET,PVT=1000
```

Response from the z/OS system console is as follows:

```
DXR177I IR21033 THE VALUE FOR PVT IS SET TO 1000
```

Example 6: Enter the following command on a z/OS console:

```
 F IR21I,SET,MLT=4G
```

Response from the z/OS system console is as follows:

```
DXR177I IR21033 THE VALUE FOR MLT IS SET TO 4G
```

Chapter 51. MODIFY irlmproc,STATUS (z/OS IRLM)

This command displays information for one or more subsystems connected to the IRLM that is specified using *irlmproc*. Each subsystem connected to the specified IRLM is listed, including subsystem name, status, work unit, lock information, the current values of the IRLM TIMEOUT, and DEADLOCK parameter values. Additionally, you can list an IRLM's ID and service level. For a specified IRLM, you can display the current storage allocated, as well as the greatest amount of storage that was allocated since the last time this IRLM was started.

Abbreviation: F

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage notes" on page 289
- "Examples" on page 289

Environment

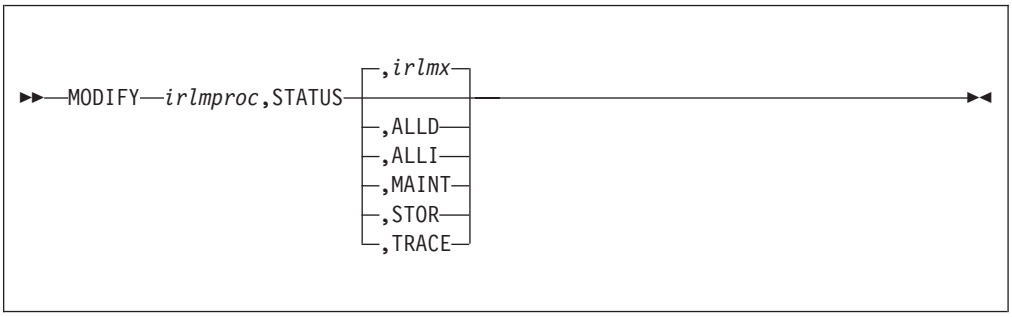
This command can be issued only from a z/OS console.

Data sharing scope: Member or group, depending on which option you choose

Authorization

The command requires an appropriate level of z/OS authority, as described in *z/OS MVS System Commands.*

Syntax



Option descriptions

irlmproc
Specifies the IRLM that is to process the command.

© Copyright IBM Corp. 1983, 2008

irlmx
Specifies which IRLM's status is to be displayed. *irlmx* is the concatenation of the IRLM subsystem name and IRLM member ID as specified in the IRLM startup procedure (DB2 installation panel DSNTIPI). An example is DJ2A2 (the member ID is 2).

ALLD
Requests the DB2 subsystem name and status of a DB2 that is identified to an IRLM. In a data sharing group, this command lists information about all DB2 subsystems that are currently identified to an IRLM, assuming that the IRLM on which the command is issued is connected to the data sharing group. You can determine if the IRLM is connected by issuing a MODIFY irlmproc,STATUS command and checking that the output shows SCOPE=GLOBAL.

If a DB2 is down and holds retained locks, that DB2 is also displayed. However, the IRLM that is displayed with that DB2 can vary depending on several circumstances:

- Normally, it is the last IRLM to which the DB2 subsystem identified.
- If a rebuild of the lock structure occurred after the retained locks were created, the IRLM with the lowest member ID at the time the rebuild occurred is displayed.
- If a group restart is occurring and one DB2 subsystem is recovering on behalf of another DB2 subsystem, the IRLM that is displayed is the one associated with the DB2 subsystem doing the peer recovery. For example, if DB1A is doing a peer recovery of DB2A, the display might show the following information:

```
NAME  STATUS ... IRLM_NAME
DB1A  UP         IRLA
DB2A  DOWN       IRLA
```

ALLI
Requests the IRLM subsystem name, ID, status, and service level. In a data sharing group, this command lists information about all IRLM subsystems in the data sharing group, assuming that the IRLM on which the command is issued is connected to the data sharing group. You can determine if the IRLM is connected by issuing a MODIFY irlmproc,STATUS command and checking that the output shows SCOPE=GLOBAL.

If an IRLM is down, it is displayed only if its associated DB2 subsystem is down and holds retained locks. The IRLM that is displayed can vary depending on several circumstances:

- Normally, it is the last IRLM to which the DB2 subsystem identified.
- If a rebuild of the lock structure occurred after the retained locks were created, the IRLM with the lowest member ID at the time the rebuild occurred is displayed.
- If the failed DB2 subsystem had recovery done on its behalf by another DB2 subsystem, the IRLM that is displayed is the one associated with the DB2 subsystem that did the peer recovery.

MAINT
For this IRLM only, displays the maintenance levels of IRLM load module CSECTS in a two-column format.

STOR
For this IRLM only, displays the current and maximum allocation for CSA, ECSA, and private extended storage.

TRACE
Requests information about IRLM subcomponent trace types. Information includes whether a subcomponent trace type is active, how many trace buffers are used by the trace, and whether the component trace external writer is active for the trace.

Usage notes

Messages: If *irlmx* is not specified, or if this IRLM is in a non-data-sharing environment, message DXR101I is issued. That message lists each subsystem connected to the IRLM specified by *irlmx*, with an indication as to whether the connection is active.

Displaying IRLM IDs: If *irlmproc* is started specifying SCOPE=GLOBAL, the second line of the display indicates the IRLM IDs of the IRLM subsystems.

Examples

Example 1: Enter the following command on the z/OS system console:

```
MODIFY IRTPROC,STATUS
```

Response on the z/OS system console:

```
DXR101I IR2T001 STATUS SCOPE=LOCAL
DEADLOCK: 0500
        SUBSYSTEMS IDENTIFIED                PT01
        NAME      T/OUT     STATUS    UNITS     HELD      WAITING     RET_LKS
        DSNT1     0010      UP-NS     0005      0010      0002        0
```

Explanation: The operator on the z/OS system has requested information about the DB2 systems connected to the IRLM identified by the IRLM procedure named IRTPROC.

If the IRLM is SCOPE=GLOBAL on the irlmproc and is not connected to any group, the status message shows:

```
DXR101I IR21001 STATUS SCOPE=DISCON
```

Other information includes:

DEADLOCK
IRLM's deadlock detection interval time. he value is measured in milliseconds.

T/OUT
The timeout value specified by DBMS for resource timeout. The value is measured in seconds.

STATUS
Indicates the status of the DB2 subsystem:

UP — The value UP in the STATUS field indicates that the DB2 subsystem is active.

DOWN — The value DOWN in the STATUS field indicates that the DB2 subsystem is failed. All modify type locks held by this DB2 subsystem have been retained by IRLM. The DB2 subsystem is known to be down only if it holds retained locks.

RET_LKS
The number of retained locks that are owned by this DB2 subsystem.

Example 2: Assume that you have a data sharing group. Enter the following command on the system console:

```
MODIFY DB1GIRLM,STATUS,ALLD
```

Response on system console is as follows:

```
11.11.07 STC00061  DXR102I DJ1G001 STATUS                              C
        SUBSYSTEMS IDENTIFIED
        NAME     STATUS    RET_LKS   IRLMID  IRLM_NAME  IRLM_LEVL
        DB1G     UP              0    001      DJ1G        2.022
        DB2G     UP              0    002      DJ2G        1.022
    DXR102I End of display
```

Explanation: The output shows all the DB2 subsystems that are connected to IRLMs in this data sharing group (the group to which the IRLM processing the request belongs). Other information includes:

STATUS
Indicates the status of the DB2 subsystem:

UP
The value UP in the STATUS field indicates that the DB2 subsystem is active.

DOWN
The value DOWN in the STATUS field indicates that the DB2 subsystem is failed. All modify type locks held by this DB2 subsystem have been retained by IRLM. The DB2 subsystem is known to be down only if it holds retained locks.

SYSFAIL
The value SYSFAIL in the STATUS field indicates that the IRLM that this DB2 subsystem is identified to is disconnected from the data sharing group. All modify type lock that are held by this DB2 subsystem are retained by IRLM. The DB2 subsystem is known to be SYSFAIL only if it holds retained locks.

RET_LKS
The number of retained locks that are owned by this DB2 subsystem.

IRLMID
The ID of the IRLM to which this DB2 subsystem is identified.

IRLM_NAME
The name of the IRLM to which this DB2 subsystem is identified.

IRLM_LEVL
The IRLM release and function level that this DB2 subsystem requested on the identify to IRLM. This is in the form of *r.fff*.

Example 3: To display information about a specific member of a data sharing group, enter the following command:

```
MODIFY DB1GIRLM,STATUS,DJ1G002
```

Response on system console is as follows:

```
11.11.21 STC00061 DXR102I DJ1G001 STATUS                            C
        SUBSYSTEMS IDENTIFIED
        NAME     STATUS    RET_LKS  IRLMID  IRLM_NAME   IRLM_LEVL
        DB1G     UP              0   002      DJ1G         2.022
     DXR102I End of display
```

Explanation: This output shows information similar to the output that is shown in example 1, but this command specifies a specific IRLM in the data sharing group.

Example 4: Again, assume data sharing is in effect. Enter the following command on the system console:

```
MODIFY DB1GIRLM,STATUS,ALLI
```

The response on the console is as follows:

```
11.11.00 STC00061  DXR103I DJ1G001 STATUS                          C
        IRLMS PARTICIPATING IN DATA SHARING GROUP FUNCTION LEVEL=1.022
        IRLM_NAME IRLMID STATUS LEVEL  SERVICE  MIN_LEVEL MIN_SERVICE
          DJ1G      001    UP   2.022  HIR2220    2.022     HIR2220
          DJ2G      002    UP   1.022  PQ52360    1.012     PN90337
    DXR103I End of display
```

Explanation: The output shows the IRLMs that are participating in this data sharing group (the group which includes the IRLM processing the request). Other information includes:

STATUS
The value "UP" in the STATUS field indicates that the IRLM is active. STATUS shows "DOWN" if the IRLM is failed. An IRLM is known to be "DOWN" only if the DB2 subsystem that was identified to it holds retained locks. This connection between a failed DB2 subsystem and IRLM is lost after a REBUILD or a group restart.

LEVEL
The current IRLM release and function level in the form of *r.fff*.

SERVICE
The IRLM service or release that corresponds to the release and function level that is given in ″LEVEL″.

MIN_LEVEL
The minimum IRLM function level with which this IRLM can coexist.

MIN_SERVICE
The IRLM service or release that corresponds to the function level given in ″MIN-LEVEL.″

Group Function Level
The IRLM release and function level that is in use by all the IRLMs in this data sharing group.

Example 5: Assume that this command is issued in a non-data-sharing environment. Enter the following command on the system console:

```
MODIFY DB1GIRLM,STATUS,ALLI
```

The response on the console is as follows:

```
11.11.03 STC00082  DXR103I DJ1G001 STATUS                            C
        IRLMS PARTICIPATING IN DATA SHARING GROUP FUNCTION LEVEL=2.022
        IRLM_NAME IRLMID STATUS LEVEL SERVICE MIN_LEVEL MIN_SERVICE
          DJ1G      001    UP   2.022 HIR2220   1.022     PQ523690
     DXR103I End of display
```

Explanation: The output shows information only for the specified IRLM. The group function level that is shown is the function level for the specified IRLM. Refer to Example 3 on page 291 for additional information about interpreting output.

Example 6: Enter the following command on the system console:

```
MODIFY IR21PROC,STATUS,STOR
```

The response on the console is as follows:

```
DXR100I PR21001 STOR STATS 150
        PC: YES     LTEW: 32  LTE:    OM RLE:  3046     RLEUSE: 0
     BB PVT:  1738M  AB PVT (MEMLIMIT):    2G
   CSA USE: ACNT:  OK  AHWM:  OK  CUR:  312K  HWM:  312K
   ABOVE 16M      15  312K    BELOW 16M:      0      0K
CLASS   TYPE  SEGS     MEM   TYPE   SEGS     MEM   TYPE  SEGS     MEM
ACCNT   T-1      1   2048K   T-2       1    1024K   T-3      1      4K
PROC    WRK      4     20K   SRB       1       1K   OTH      1      1K
MISC    VAR      7    733K   N-V      10    247K    FIX      1     24K
DXR100I  END OF DISPLAY
```

Explanation: The example shows that current storage allocated for IRLM is 312 KB, and the greatest amount that has been allocated since the last time IRLM was started is also 312 KB. The storage for the locking structures (RHB and RLB) is contained within IRLM private storage. Use the following information to interpret the display output:

PC Displays the current value for the PC option of the IRLM startup procedure. For DB2 Version 8, this value will always be YES.

BB PVT (MEMLIMIT)
Displays the current amount of private storage below the bar. The PVT value is 1738 MB in this example.

AB PVT (MEMLIMIT)
Displays the current limit to private storage above the bar, also known as the MEMLIMIT, which is managed by MVS. This storage is used for locks. The AB PVT value is 2 GB in this example.

LTE The number of lock table entries that were available in the coupling facility the last time this IRLM was connected to the group. Each unit consists of 1 048 576 entries. If LTE is less than one unit, the value will be zero.

LTEW The lock table entry width is the number of bytes needed for each lock table entry in the lock structure. The more users in the group, the more bytes are needed to manage each lock table entry. For up to 6 users, the LTEW is 2, for up to 22 users, the LTEW is 4, and for up to 32 users, the LTEW is 8.

RLE The number of record table entries that were available in the coupling facility the last time this IRLM was connected to the group.

RLEUSE The number of RLE that are in use in the coupling facility at the time you issue the MODIFY command. If the IRLM is disconnected from the CF, this number represents the RLE that were in use when the IRLM last updated prior to DISCONNECT.

CSA USE CSA USE is unused in DB2 Version 8 and is displayed for compatibility reasons only.

CUR Shows the total current CSA and ECSA usage. In this case, the current usage (CUR) is 312 KB, and the high water mark (HWM) is also 312 KB. The accountable storage is a subset of this total storage.

ACCNT The ACCNT row of the report is a breakdown of lock control block structures and their storage use.

T-1 Type one structures are for resources. In this case, it shows that one storage segment is held for a total of 2048 KB.

T-2 Type two structures are for all resource requests after the first request for a specific resource. This example shows that one storage segment is held for a total of 1024 KB.

T-3 Type three structures are for requesters or work units that are waiting for or hold resources. This example shows that one storage segment is held for a total of 4 KB.

PROC and MISC rows

These rows contain usage information for private storage that is used to process DBMS requests. Use this information under the guidance of IBM Software Support for diagnosing problems.

For more information, see the explanation of message DXR100I in *DB2 Codes*.

Example 7: Enter the following command on the system console:

```
MODIFY PR21PROC,STATUS,TRACE
```

The command displays the following output on the system console:

```
DXR179I PR21034 TRACE USAGE
TRACE BUFFER STORAGE IN USE:   256 KB
MAXIMUM NUMBER OF TRACE BUFFERS ALLOWED PER TRACE TYPE:   10
TRACE TYPE   ACTIVE   BUFFERS IN USE   CTRACE WRITER
----------   ------   --------------   -------------
   SLM          N            0              N
   XIT          Y            2              N
   XCF          N            0              N
   DBM          N            0              N
   EXP          Y            1              N
   INT          Y            1              N
```

Explanation: This example shows that the storage currently allocated for IRLM tracing is 256 KB, the maximum number of trace buffers allowed per trace type is set to 10, and the external CTRACE writer is not active. For more information about the trace types, see Chapter 89, "TRACE CT (z/OS IRLM)," on page 433.

Use the z/OS TRACE CT command, described in Chapter 89, "TRACE CT (z/OS IRLM)," on page 433 to activate or deactivate traces. You cannot turn off the EXP and INT traces. The XIT (for data sharing), EXP, and INT traces are automatically activated when you start IRLM. All traces are automatically activated with IRLMPROC TRACE=YES.

The trace size for each buffer is 64 KB. Use the MODIFY irlmproc,SET,TRACE=*nnn* command on page 283 to change the maximum number of trace buffers.

Example 8: Enter the following command on the system console:

```
MODIFY IR21I,STATUS,MAINT
```

The command displays the following output on the system console:

```
 DXR104I IR21240 MAINTENCE LEVELS
     LMOD.Csect     MaintLv   Date       Csect    APAR     DATE
 DXRRLM00.DXRRL010  PQ35083 02/22/00    DXRRL020  PQ35083 02/22/00
          DXRRL030  PQ27464 08/18/99    DXRRL040  PQ35083 02/22/00
```

Explanation: The output shows the maintenance levels of IRLM load module CSECTS in a two-column format.

Chapter 52. -MODIFY TRACE (DB2)

The DB2 command MODIFY TRACE does the following:
- Changes the trace events (IFCIDs) being traced for a particular active trace.
- Stops any IFCID previously active for the specified trace.
- Writes statistics records.

Abbreviation: -MOD TRA

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 296
- "Option descriptions" on page 296
- "Example" on page 297

Environment

This command can be issued from a z/OS console, a DSN session, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

Traces started by a IFI/IFC program: Before you modify an active trace, ensure that an IFI application program or the IFC Selective Dump utility (DSN1SDMP) did not start the trace. If you modify a trace started by DSN1SDMP, the DSN1SDMP utility abnormally terminates. When DSN1SDMP terminates, it stops the trace. This stop could interfere with the MODIFY TRACE command, which stops and restarts the trace.

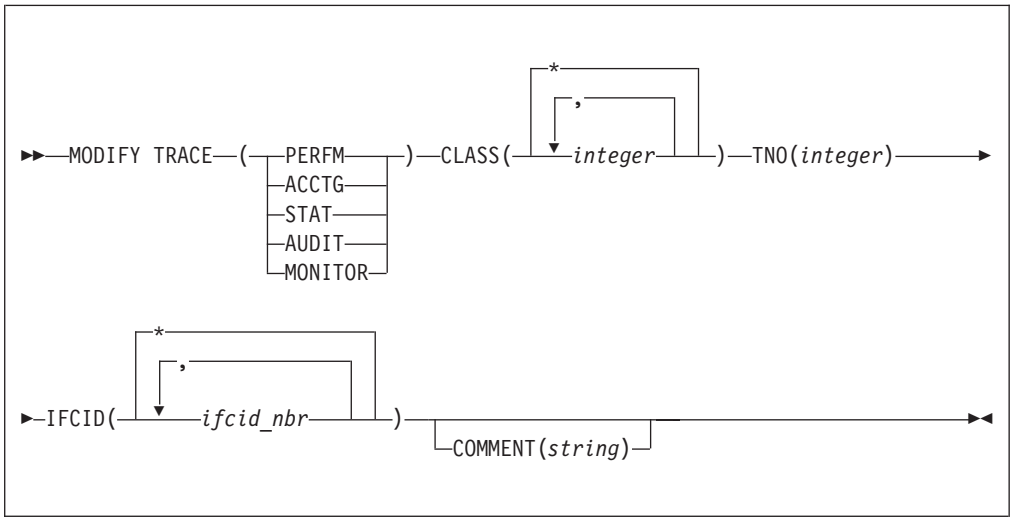
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:
- TRACE privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

TRACE
Determines which IFCIDs are started. Table 27 lists each trace type, its abbreviation, and a brief description of each type. For more information about each trace type, refer to Chapter 77, "-START TRACE (DB2)," on page 383.

Table 27. Trace types

Type	Abbreviation	Description
PERFM	P	Performance records of specific events
ACCTG	A	Accounting records for each transaction
STAT	S	Statistical data
AUDIT	AU	Audit data
MONITOR	MON	Monitor data

One additional trace type is not described here. It is intended for service and is to be used under the direction of IBM Software Support. For details, see *DB2 Diagnosis Guide and Reference*.

CLASS(*integer*, ...)
Limits the list to IFCIDs started for specified classes.

Abbreviation: C

integer is a class to which the list of IFCIDs started is limited. For descriptions of the allowable classes, see Chapter 77, "-START TRACE (DB2)," on page 383.

The **default** is **CLASS(*)**, which starts all default IFCID classes.

TNO(*integer*)
Specifies the particular trace to be modified, identified by its trace number (1 to 32, 01 to 09). You can specify only one trace number. TNO is a required option for the MODIFY TRACE command.

No default exists for the TNO keyword.

IFCID(*ifcid_nbr*, **...)**
Specifies which other IFCIDs (trace events), in addition to those IFCIDs contained in the classes specified in the CLASS option, are to be started. To start only those IFCIDs specified in the IFCID option, use trace classes 30-32. These classes have no predefined IFCIDs and are available for a location to use. (See the example on page 297 for an example of activating only those trace events specified in the IFCID option.)

If you do not specify the IFCID option, only those IFCIDs contained in the activated trace classes are started.

The maximum number of IFCIDs is 156. The range of values that are valid for the IFCID option is 1 through 350, with the exception of: 4, 5, 185, 187, 217, 232, 234, 240, and 241.

The **default** is **IFCID(*).**

COMMENT(*string***)**
Specifies a comment that is reproduced in the trace output record (except in the resident trace tables).

string is any character string; it must be enclosed between apostrophes if it includes a blank, comma, or special character.

Example

Change trace number six so that it collects only statistics and accounting data. You can define CLASS(30) at your site.

```
-MODIFY TRACE(S) IFCID(1,2,3) TNO(6) CLASS(30)
  COMMENT ('STATS AND ACCOUNTING ON')
```

Chapter 53. REBIND PACKAGE (DSN)

The DSN subcommand REBIND PACKAGE rebinds an application package when you make changes that affect the package, but have not changed the SQL statements in the program. For example, you can use REBIND PACKAGE when you change the authorizations, create a new index for the package, or use RUNSTATS. When the REBIND PACKAGE(*) command is issued, trigger packages will not be affected.

REBIND PACKAGE is generally faster and more economical than BIND PACKAGE. You should use BIND PACKAGE with the ACTION(REPLACE) option under the following conditions:

- When you change the SQL statements
- When you recompile the program
- When you have previously run BIND PACKAGE with the SQLERROR(CONTINUE) option

For more information about using REBIND PACKAGE, see Part 5 of *DB2 Application Programming and SQL Guide*.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 301
- "Option descriptions" on page 302
- "Usage note" on page 302
- "Example" on page 302

Environment

You can use REBIND PACKAGE through DB2I, or enter the REBIND PACKAGE subcommand from a DSN session running in foreground or background.

Data sharing scope: Group

Authorization

The package owner must have authorization to execute **all** SQL statements embedded in the package for REBIND PACKAGE to build a package without producing error messages. For VALIDATE(BIND), DB2 verifies the authorization at bind time. For VALIDATE(RUN), DB2 verifies the authorization initially at bind time, but if the authorization check fails, DB2 rechecks it at run time.

Table 28 on page 300 explains the authorization required to run REBIND PACKAGE, depending on the options specified.

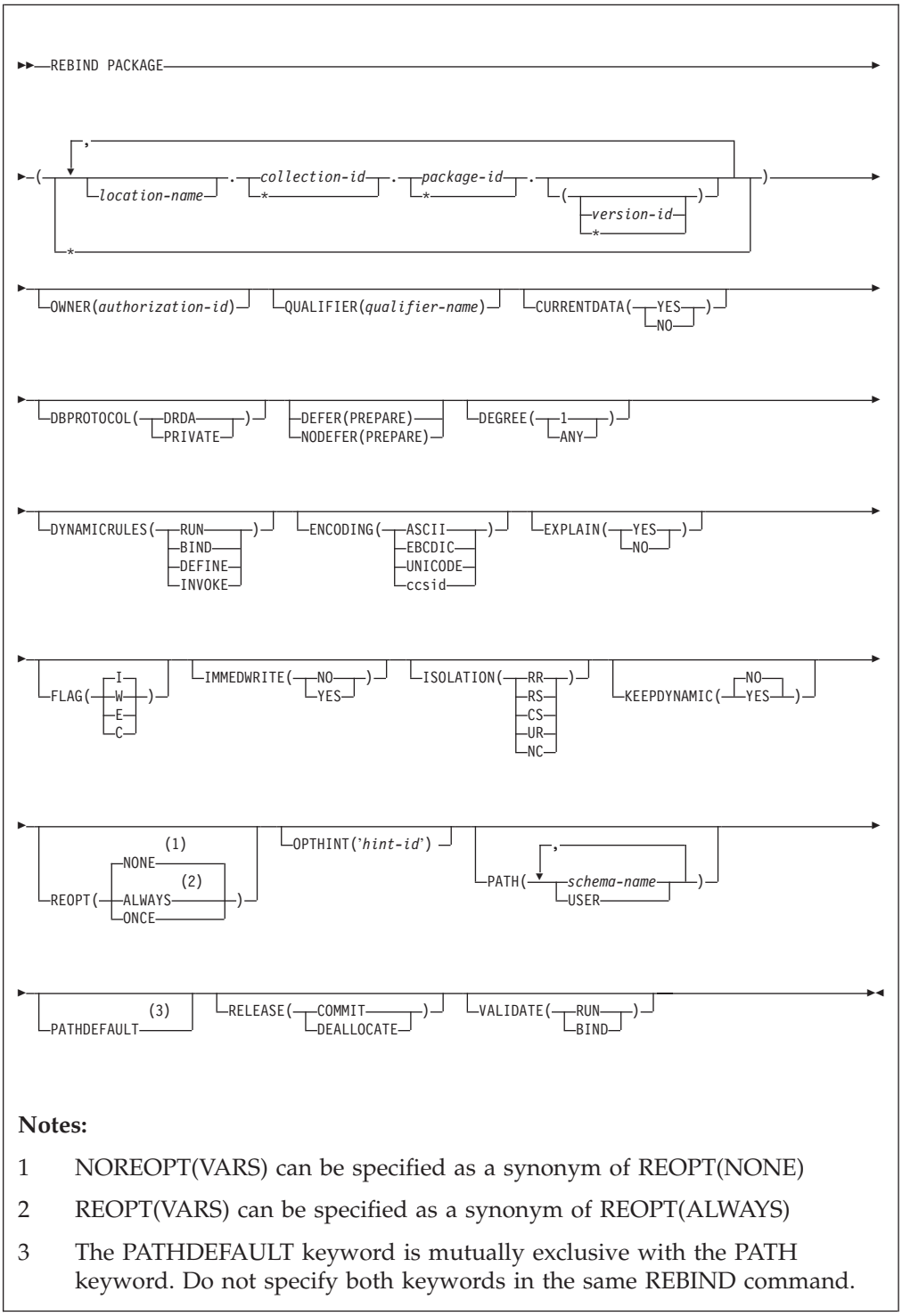
© Copyright IBM Corp. 1983, 2008

Table 28. Summary of privileges for REBIND PACKAGE

Option	Authorization required to run REBIND PACKAGE
REBIND PACKAGE with no change in ownership because the OWNER keyword is not specified.	The authorization IDs of the process must have one of the following authorities: • Ownership of the package • BIND privilege on the package • BINDAGENT privilege from the owner of the package • PACKADM authority on the collection or on all collections • SYSADM or SYSCTRL authority
REBIND PACKAGE with no change in ownership, although the original owner is specified for the OWNER keyword.	The authorization IDs of the process must have one of the following authorities: • OWNER *authorization-id* must be one of the primary or secondary authorization IDs of the binder • BINDAGENT privilege from the owner of the package • SYSADM or SYSCTRL authority
REBIND PACKAGE with change of ownership. (An authorization ID that is not the original owner is specified in the OWNER keyword.)	The new OWNER must have one of the following authorities: • BIND privilege on the package • PACKADM authority on the collection or on all collections • SYSADM or SYSCTRL authority **Specifying the OWNER:** If any of the authorization IDs have the BINDAGENT privilege granted from the owner, the *authorization-id* can specify the grantor as OWNER. Otherwise, OWNER *authorization-id* must be one of the primary or secondary authorization IDs of the binder.

For additional information about the authorization required to execute BIND PLAN, see Part 5 (Volume 2) of *DB2 Administration Guide*.

Syntax



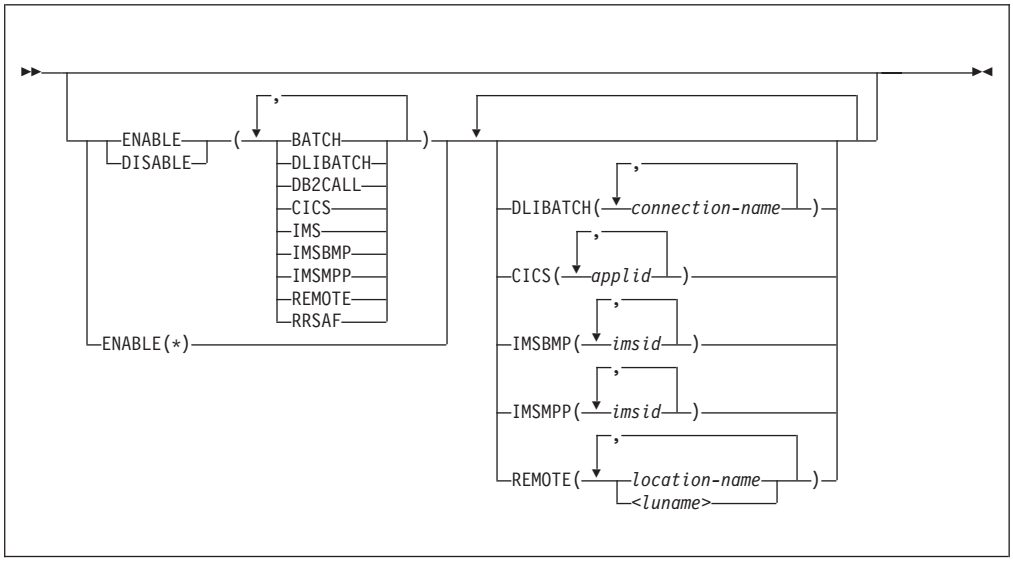
Notes:

1 NOREOPT(VARS) can be specified as a synonym of REOPT(NONE)

2 REOPT(VARS) can be specified as a synonym of REOPT(ALWAYS)

3 The PATHDEFAULT keyword is mutually exclusive with the PATH keyword. Do not specify both keywords in the same REBIND command.

enable-block:



Option descriptions

For descriptions of the options shown in the syntax diagram, refer to Chapter 15, "BIND and REBIND options," on page 63.

Usage note

If you rebind multiple packages, DB2 commits each successful rebind before rebinding the next package.

Example

Rebind packages TEST.DSN8BC81.(MAY_VERSION) and PRODUCTION.DSN8BC81.(DEC_VERSION), both of which are located at the local location USIBMSTODB22. The packages can run only from the CICS or the DLIBATCH environments if the connection ID is CON2. This replaces the CON1 that is specified on the BIND PACKAGE command.

```
REBIND PACKAGE (USIBMSTODB22.TEST.DSN8BC81.(MAY_VERSION),
                USIBMSTODB22.PRODUCTION.DSN8BC81.(DEC_VERSION)) -
  ENABLE (CICS,DLIBATCH) CICS (CON2)
```

Chapter 54. REBIND PLAN (DSN)

The DSN subcommand REBIND PLAN rebinds an application plan when you make changes that affect the plan, but do not change the SQL statements in the programs. For example, you can use REBIND PLAN when you change authorizations, create a new index for the plan, or use RUNSTATS. If the rebind is successful, the process prepares an application plan and updates its description in the catalog table SYSPLAN.

REBIND PLAN is generally faster and more economical than BIND PLAN. But if you change the SQL statements or recompile a program, you should use BIND PLAN with the option ACTION(REPLACE).

For more information about using REBIND PLAN, refer to Part 5 of *DB2 Application Programming and SQL Guide*.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 305
- "Option descriptions" on page 306
- "Usage note" on page 306
- "Example" on page 306

Environment

You can use REBIND PLAN through DB2I, or enter the REBIND PLAN subcommand from a DSN session running in foreground or background.

Data sharing scope: Group

Authorization

The plan owner must have authorization to execute **all** SQL statements embedded in the plan for REBIND PLAN to build a plan without producing error messages. For VALIDATE(BIND), DB2 verifies the authorization at bind time. For VALIDATE(RUN), DB2 initially verifies the authorization at bind time, but if the authorization check fails, DB2 rechecks it again at run time. If you use the PKLIST keyword, you must have EXECUTE authority for the packages or collections specified on PKLIST.

Table 29 on page 304 explains the authorization required to run REBIND PLAN, depending on the options specified.

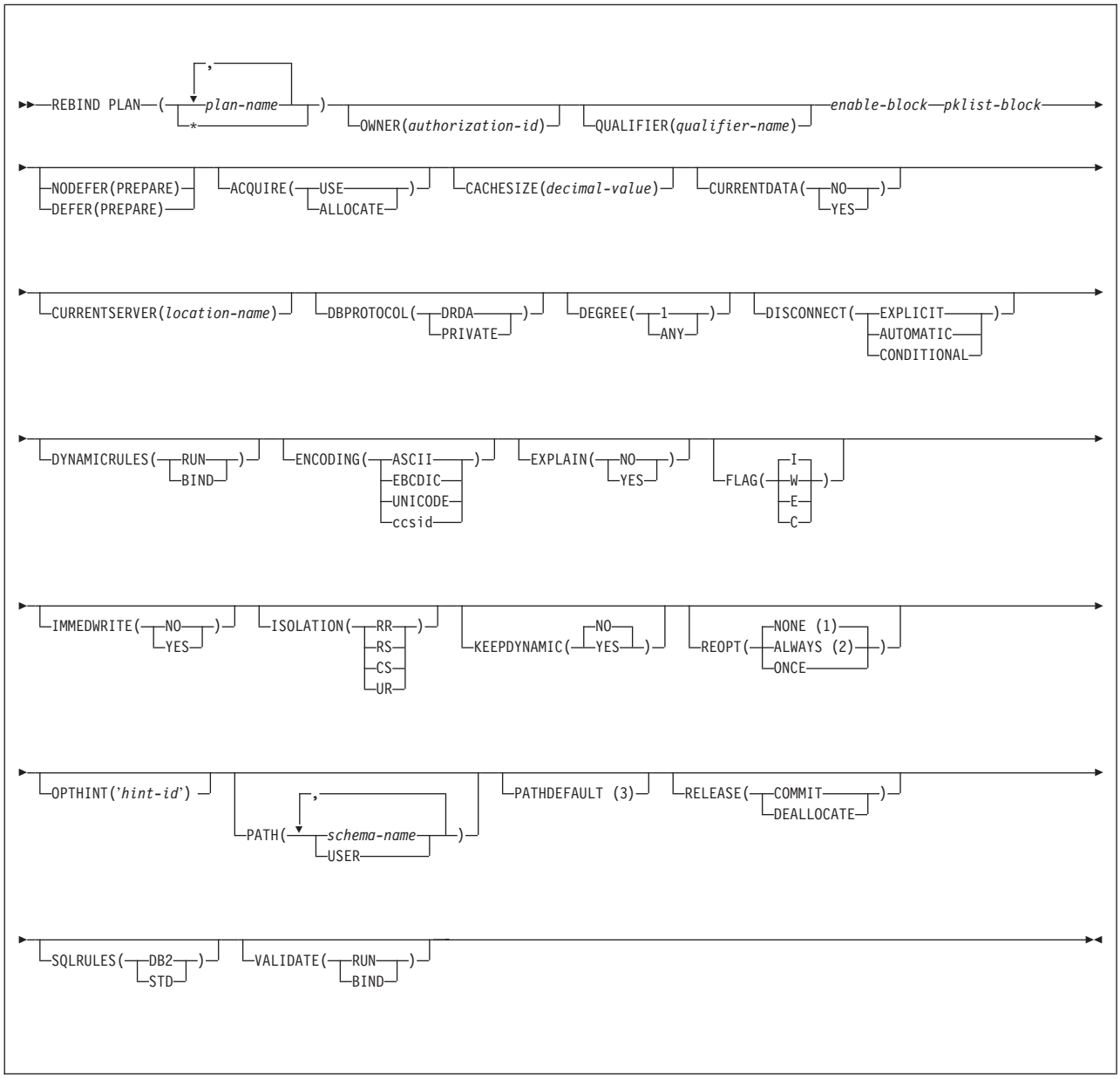
© Copyright IBM Corp. 1983, 2008

Table 29. Summary of privileges for REBIND PLAN

Option	Authorization required to run REBIND PLAN
REBIND PLAN with no change in ownership because the OWNER keyword is not specified.	The authorization IDs of the process must have one of the following authorities: • Ownership of the plan • BIND privilege on the plan • BINDAGENT privilege from the owner of the plan • SYSADM or SYSCTRL authority
REBIND PLAN with no change in ownership, although the original owner is specified for the OWNER keyword.	The authorization IDs of the process must have one of the following authorities: • OWNER *authorization-id* must be one of the primary or secondary authorization IDs of the binder • BINDAGENT privilege from the owner of the plan • SYSADM or SYSCTRL authority
REBIND PLAN with change of ownership. (An authorization ID that is not the original owner is specified in the OWNER keyword.)	The new OWNER must have one of the following authorities: • BIND privilege on the plan • SYSADM or SYSCTRL authority **Specifying the OWNER:** If any of the authorization IDs has the BINDAGENT privilege granted from the owner, then *authorization-id* can specify the grantor as OWNER. Otherwise, OWNER *authorization-id* must be one of the primary or secondary authorization IDs of the binder.
PKLIST, specifying individual packages	Authorization ID of the process must include one of the following authorities: • EXECUTE privilege on each package specified in the PKLIST • PACKADM authority on specific collections containing packages or on collection * • SYSADM authority
PKLIST, specifying (*), indicating all packages in the collection	Authorization ID of the process must include one of the following authorities: • EXECUTE privilege on each package in the collection • EXECUTE privilege on *collection-id*.* • PACKADM authority on *collection-id* or on * • SYSADM authority

For additional information about the authorization that is required to execute REBIND PLAN, see Part 5 (Volume 2) of *DB2 Administration Guide*.

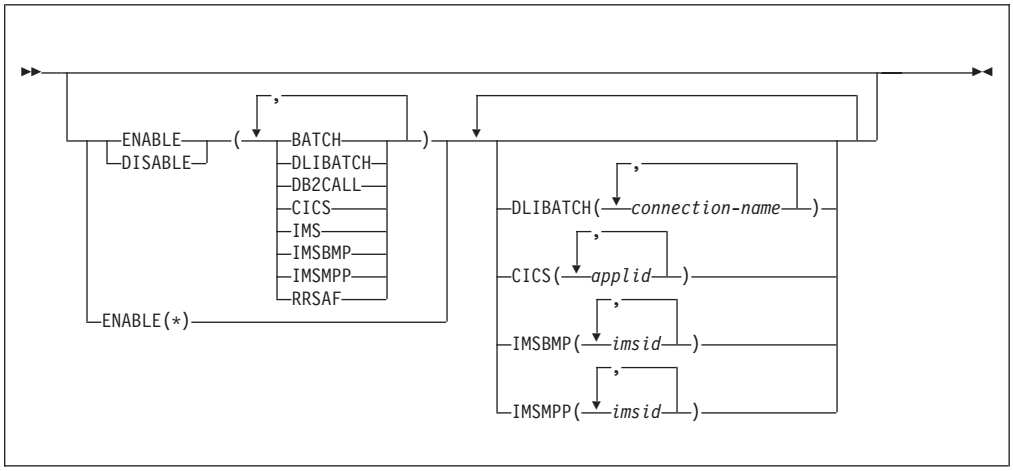
Syntax



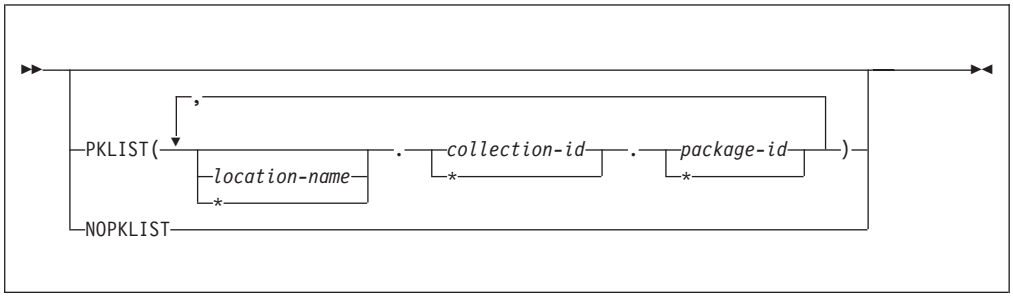
Notes:

1. REOPT(VARS) can be specified as a synonym of REOPT(ALWAYS)
2. NOREOPT(VARS) can be specified as a synonym of REOPT(NONE)
3. The PATHDEFAULT keyword is mutually exclusive with the PATH keyword. Do not specify both keywords in the same REBIND command.

enable-block:



pklist-block:



Option descriptions

For descriptions of the options shown in the syntax diagram, refer to Chapter 15, "BIND and REBIND options," on page 63.

Usage note

If you rebind multiple plans, DB2 commits each successful rebind before rebinding the next plan.

Example

Rebind plan DSN8BC81 to enable DB2 to take advantage of a newly created index. Use FLAG(W) to issue warning, error, and completion messages, but not informational messages. Use VALIDATE(BIND) to point out any error conditions during the bind process. Use ISOLATION(CS) to prevent other applications from changing the database values that this application uses only while the application is using them. This isolation level protects changed values until the application commits or terminates. Omit the OWNER keyword to leave the plan's owner authorization ID the same. Omit the ENABLE or DISABLE keywords to use the connections previously defined for the plan.

```
REBIND PLAN (DSN8BC81) -
  FLAG (W) -
  VALIDATE (BIND) -
  ISOLATION (CS)
```

Chapter 55. REBIND TRIGGER PACKAGE (DSN)

The DSN subcommand REBIND TRIGGER PACKAGE rebinds a package that was created when DB2 executed a CREATE TRIGGER statement. You can use this subcommand to change a limited subset of the default bind options that DB2 used when creating the package. You might also rebind a trigger package to re-optimize its SQL statements after you create a new index or use the RUNSTATS utility. Additionally, you can rebind a trigger package if it has been marked invalid because an index, or another object it was dependent on, was dropped.

If the rebind is successful, the trigger package is marked valid. When REBIND TRIGGER PACKAGE(*) is issued, the rebind will affect all trigger packages that the issuer is authorized to rebind. Trigger packages cannot be rebound remotely. The location name is permitted when specifying the package name on a REBIND TRIGGER PACKAGE subcommand. However, the location name must not refer to a remote location.

For more information about using REBIND TRIGGER PACKAGE, see Part 5 of *DB2 Application Programming and SQL Guide*.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 308
- "Option descriptions" on page 308
- "Usage notes" on page 309
- "Output" on page 309
- "Example" on page 309

Environment

You can use REBIND TRIGGER PACKAGE through DB2I, or enter the REBIND TRIGGER PACKAGE subcommand from a DSN session that is running in foreground or background.

Data sharing scope: Group

Authorization

To build a package without producing error messages, the package owner must have authorization to execute **all** SQL statements that are embedded in the package for REBIND TRIGGER PACKAGE.

To execute this subcommand, you must use a privilege set of the process that includes one of the following privileges or authorities:

- Ownership of the trigger package
- BIND privilege on the trigger package
- BINDAGENT privilege from the owner of the trigger package
- PACKADM authority on the collection or on all collections
- SYSADM authority
- SYSCTRL authority

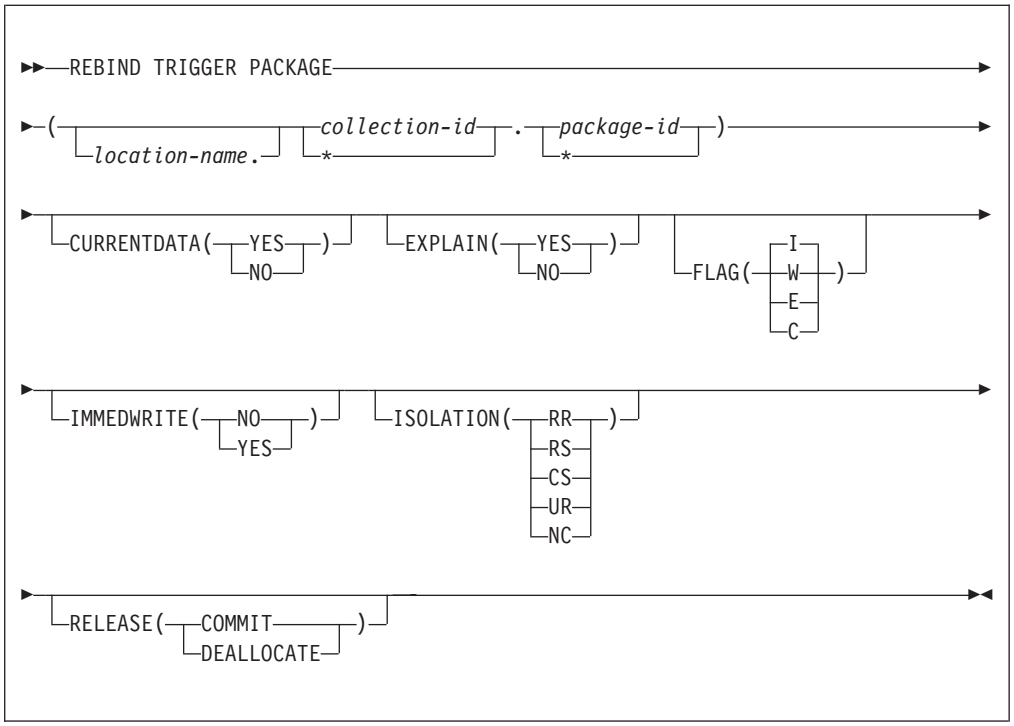
© Copyright IBM Corp. 1983, 2008

When the trigger package is bound, the privileges of the current authorization ID are used when checking authority to bind statements within the triggered action. On REBIND TRIGGER PACKAGE, you need one of the following privileges or authorities:

- Ownership of the package
- BIND privilege on the package
- BINDAGENT privilege granted from owner
- PACKADM authority
- SYSADM authority
- SYSCTRL authority

For additional information about the authorization required to execute REBIND TRIGGER PACKAGE, see Part 5 (Volume 2) of *DB2 Administration Guide*.

Syntax



Option descriptions

TRIGGER PACKAGE

Determines what trigger package or packages to rebind.

The following options identify the location, collection, and package name of the package. You can identify a location and collection. For REBIND TRIGGER, you must identify a trigger package name.

location-name

Identifies the current local location. Remote rebind of a trigger package is not allowed. *location-name* is the location of the DBMS where the package rebinds and where the description of the package resides.

The **default** is the local DBMS.

collection-id or *
Identifies the schema-name that already contains the trigger package to rebind. No default exists.

For REBIND TRIGGER, you can use an asterisk (*) to rebind all local packages with the specified *package-id* in all the collections for which you have bind privileges.

package-id or *
Identifies the name of the trigger package to rebind, as listed in the NAME column of the SYSPACKAGE catalog table. No default exists.

You can use the pattern-matching character (*) to rebind all local triggers in *collection-id* for which you have bind privileges.

For descriptions of the options that are shown in the syntax diagram, see Chapter 15, "BIND and REBIND options," on page 63.

For more information about specifying schema names and trigger packages for the REBIND TRIGGER PACKAGE command, see Part 4 of *DB2 Application Programming and SQL Guide*.

Usage notes

Restrictions on trigger packages: A trigger package can be explicitly rebound, but it cannot be explicitly bound using the BIND PACKAGE subcommand.

A trigger package cannot be explicitly freed using the FREE PACKAGE subcommand or the DROP PACKAGE statement. Use the DROP TRIGGER statement to delete the trigger package.

A trigger package cannot be copied, and it can only be rebound locally. Remote rebind of a trigger package is not allowed.

Rebinding multiple trigger packages: If you rebind multiple trigger packages, DB2 commits each successful rebind before rebinding the next package.

Restriction on trigger names:Although DBCS characters are generally allowed in
trigger names, trigger names that contain DBCS characters cannot be used in
REBIND TRIGGER PACKAGE operations.

Output

REBIND TRIGGER PACKAGE updates the COLLID and NAME columns in the SYSPACKAGE catalog table.

Example

Enter the following command to rebind trigger package TRIG1 in the ADMF001 collection of packages:

```
REBIND TRIGGER PACKAGE (ADMF001.TRIG1);
```

This command produces output that is similar to the following output:

```
DSNT254I - DSNTBRB2 REBIND OPTIONS FOR
           PACKAGE = STLEC1.ADMF001.TRIG1.()
           ACTION
           OWNER         ADMF001
           QUALIFIER     ADMF001
```

```
            VALIDATE      BIND
            EXPLAIN       NO
            ISOLATION     CS
            RELEASE       COMMIT
            COPY
DSNT255I - DSNTBRB2 REBIND OPTIONS FOR
            PACKAGE = STLEC1.ADMF001.TRIG1.()
            SQLERROR      NOPACKAGE
            CURRENTDATA   YES
            DEGREE        1
            DYNAMICRULES  BIND
            NODEFER       PREPARE
            REOPT         NONE
            KEEPDYNAMIC   NO
            DBPROTOCOL    DRDA
            QUERYOPT      1
            PATH
"SYSIBM","SYSFUN","SYSPROC","SYSADM","ADMF001"
DSNT232I - SUCCESSFUL REBIND FOR
            PACKAGE = STLEC1.ADMF001.TRIG1.()
```

Chapter 56. -RECOVER BSDS (DB2)

The DB2 command RECOVER BSDS reestablishes dual bootstrap data sets (BSDS) after one has been disabled by a data set error. Follow these steps to reestablish dual BSDS mode:

1. Use access method services to rename or delete the failing BSDS, which the DB2 system has deallocated, and define a new BSDS with the same name as the failing BSDS. You can find control statements in job DSNTIJIN.
2. Issue the DB2 command RECOVER BSDS to make a copy of the remaining BSDS in the newly allocated data set and to reestablish dual BSDS mode.

Abbreviation: -REC BSDS

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Usage note" on page 312
- "Example" on page 312

Environment

This command can be issued from a z/OS console, a DSN session, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- BSDS privilege
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax

© Copyright IBM Corp. 1983, 2008

Usage note

Using RECOVER BSDS following a BSDS I/O error: For a detailed description of the steps you must take to reestablish dual BSDS mode after a BSDS I/O error occurs, see Part 4 (Volume 1) of *DB2 Administration Guide*.

Example

Reestablish dual BSDS mode.

```
-RECOVER BSDS
```

Chapter 57. -RECOVER INDOUBT (DB2)

The DB2 command RECOVER INDOUBT recovers threads that are left in an indoubt state because DB2 or a transaction manager could not automatically resolve the indoubt status with the commit coordinator.

This command should only be used when automatic resolution will not work. The commit coordinator must determine the commit or abort decision.

Abbreviation: -REC IND

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 314
- "Option descriptions" on page 314
- "Usage note" on page 315
- "Examples" on page 316

Environment

This command can be issued from a z/OS console, a DSN session, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

Authorization

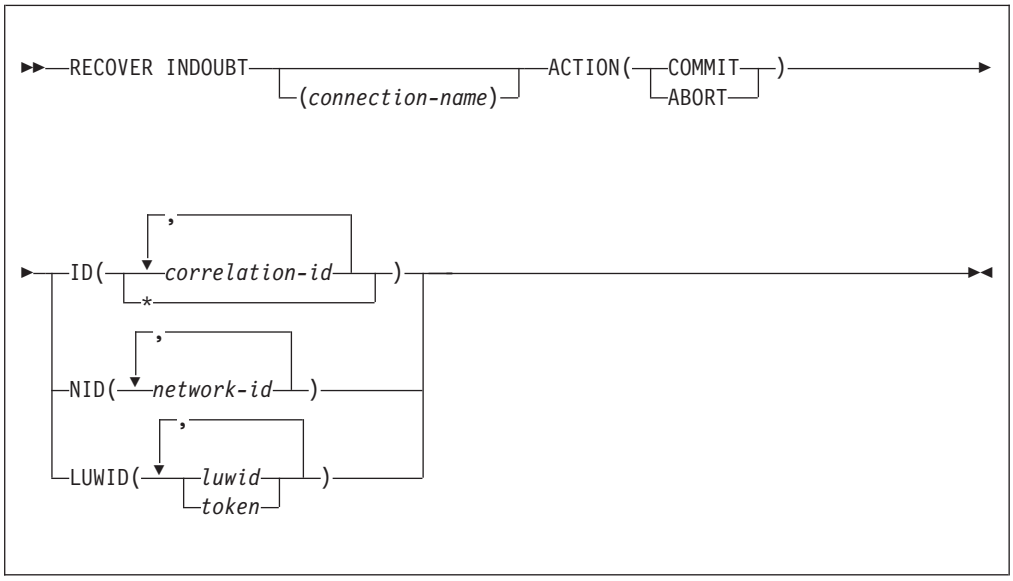
To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- RECOVER privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

(*connection-name***)**
Specifies a one- to eight-character connection name. Allied threads (including those that are distributed) that belong to the connection name are recovered. This parameter is ignored if LUWID is specified.

The **default** is the connection name from which you enter the command. If you enter this command from a z/OS console, and you are recovering an allied thread using the ID or NID parameter, you **must** supply a connection name; no default connection name is available.

ACTION
Specifies whether to commit or cancel the indoubt thread. If there are any downstream participants for which the local thread is the coordinator, the commit or abort decision is propagated to these participants.

Abbreviation: ACT

(COMMIT)
Commits the thread.

(ABORT)
Cancels the thread.

ID(*correlation-id***, ...)**
Specifies whether to recover a specific allied thread or all allied threads (including those that are distributed) that are associated with the connection name.

correlation-id
Is the correlation ID (1 to 12 characters) of a specific thread that is to be recovered. If you use more than one correlation ID, separate the IDs with commas.

Do not use a correlation ID that is associated with more than one network ID. Instead, use the NID option.

(*) Recovers all indoubt threads that are associated with the connection name. Even threads that have the same correlation ID are resolved.

NID(*network-id***, ...)**
Specifies threads to recover based on their network IDs.

network-id is a network ID that is associated with an individual thread. You can use more than one network ID for the same connection name.

For IMS and CICS connections, a network ID is specified as *net-node.number*, which is 3 to 25 characters in length.

- *net-node* is the network-node name of the system that originated the unit of work. *net-node* is one to eight characters in length.
- *number* is a unique number within the system of origin. *number* is 1 to 16 characters in length.

For RRSAF connections, a network ID is the z/OS RRS unit of recovery ID (URID) that is used to uniquely identify a unit of work. A z/OS RRS URID is a 32-character number.

The network ID appears on the recovery log of the commit coordinator as a 16-byte unique identification of a unit of work.

- For IMS and CICS, the network ID is an 8-byte node name immediately followed by an 8-byte number.
- For RRSAF connections, the network ID is a 16-byte number.

LUWID
Recovers the indoubt thread that has the specified LUWID.

luwid
Consists of an LU network name, an LUW instance number, and a commit sequence number.

The LU network name consists of a one- to eight-character network ID, a period, and a one- to eight-character network LU name. The LUW instance number consists of a period followed by 12 hexadecimal characters. The last element of the LUWID is the commit sequence number of 4 hexadecimal characters, preceded by a period.

token
A token is an alternate way to express an LUWID. DB2 assigns a token to each thread that it creates. It is a one- to six-digit decimal number that appears after the equal sign in all DB2 messages that display a LUWID.

If you enter one- to six-decimal digits, DB2 assumes that you are supplying a token. The token that DB2 assigns to a specific LUWID is unique for that DB2 subsystem, but not necessarily unique across all subsystems.

Usage note

When to use a network ID: A *network-id* is not normally needed, because a *correlation-id* can identify indoubt threads. However, if the *correlation-id* is not unique, *network-id* must be used. You do not need a *network-id* if you specify a LUWID.

If you specify a thread in the command that is part of a global transaction, the command is executed against all threads that are in the global transaction. See *DB2 Administration Guide* for an explanation of global transactions.

Examples

Example 1: Recover indoubt allied threads. Schedule a commit for all threads that are associated with the connection name from which the command is entered.

```
-RECOVER INDOUBT ACTION(COMMIT) ID(*)
```

Example 2: Recover an indoubt thread from a remote requester. Schedule a commit for the indoubt thread whose token is 1332.

```
-RECOVER INDOUBT ACTION(COMMIT) LUWID(1332)
```

Example 3: Recover indoubt threads from remote requesters. Schedule an abort for two indoubt threads. The first thread has an LUWID = DB2NET.LUNSITE0.A11A7D7B2057.0002. (The 0002 in the last segment of the LUWID represents the commit sequence number.) The second thread has a token of 442.

```
-RECOVER INDOUBT ACTION(ABORT)
         LUWID (DB2NET.LUNSITE0.A11A7D7B2057.0002, 442)
```

Chapter 58. -RECOVER POSTPONED (DB2)

The DB2 command RECOVER POSTPONED completes back-out processing for units of recovery that are left incomplete during an earlier restart (POSTPONED ABORT units of recovery). Use this command when automatic resolution is not selected.

Abbreviation: -REC POST

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions" on page 318
- "Usage note" on page 318
- "Output" on page 318
- "Example" on page 318

Environment

This command can be issued from a z/OS console, a DSN session, a DB2I panel (DB2 COMMANDS), or an IMS or CICS terminal.

Data sharing scope: Member

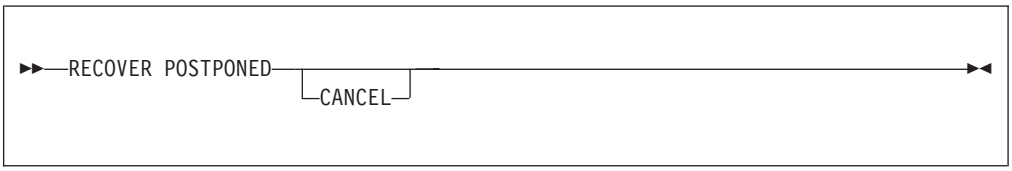
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- RECOVER privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



© Copyright IBM Corp. 1983, 2008

Option descriptions

CANCEL
Specify to stop DB2 processing of all postponed abort units of recovery immediately. Cancelling postponed abort units of recovery leaves objects in an inconsistent state.

Objects that the postponed units of recovery modify are recovered (backed out). If back out processing fails, the objects are marked as REFRESH PENDING (REFP) and either RECOVER PENDING (RECP) or REBUILD PENDING (RBDP or PSRBD) in the database exception table. Resolve the REFP status of the object by running the RECOVER utility to recover the object to a prior point in time or by running LOAD REPLACE on the object.

Usage note

Recovery action: Recovery (rollback) action is always taken for all POSTPONED ABORT units of recovery.

Output

The output from RECOVER POSTPONED consists of informational messages only.

Progression of RECOVER POSTPONED: Message DSNI024I indicates the completion of backout work against the page set or partition, and the removal of the page set or partition from the restart-pending status.

If backout processing lasts for an extended period of time, progress message DSNR047I is displayed at periodic intervals until backout processing is complete.

DB2 issues message DSN9022I after successful completion of the RECOVER POSTPONED command, or message DSN9023I if the command completed unsuccessfully. Message DSNV434I indicates that RECOVER POSTPONED was issued when no postponed-abort units of recovery needed to be resolved.

Example

Enter the following command to recover postponed-abort units of recovery.

```
-RECOVER POSTPONED
```

If postponed-abort units of recovery are found, output that is similar to the following output is generated:

```
DSNV435I - RESOLUTION OF POSTPONED ABORT URS HAS BEEN SCHEDULED
DSNI024I - DSNIARPL BACKOUT PROCESSING HAS COMPLETED
        FOR PAGESET DBKD0103.IPKD013A PART 00000004.
DSNI024I - DSNIARPL BACKOUT PROCESSING HAS COMPLETED
        FOR PAGESET DBKD0103.TPKD0103 PART 00000004.
DSNI024I - DSNIARPL BACKOUT PROCESSING HAS COMPLETED
        FOR PAGESET DBKD0103.IXKD013C PART (n/a).
DSNI024I - DSNIARPL BACKOUT PROCESSING HAS COMPLETED
        FOR PAGESET DBKD0103.IUKD013B PART (n/a).
DSNI024I - DSNIARPL BACKOUT PROCESSING HAS COMPLETED
        FOR PAGESET DBKD0103.IPKD013A PART 00000002.
DSNI024I - DSNIARPL BACKOUT PROCESSING HAS COMPLETED
        FOR PAGESET DBKD0103.TPKD0103 PART 00000002.
DSNI024I - DSNIARPL BACKOUT PROCESSING HAS COMPLETED
        FOR PAGESET DBKD0101.IXKD011C PART (n/a).
DSNI024I - DSNIARPL BACKOUT PROCESSING HAS COMPLETED
```

```
        FOR PAGESET DBKD0101.IXKD011B PART (n/a).
DSNI024I - DSNIARPL BACKOUT PROCESSING HAS COMPLETED
        FOR PAGESET DBKD0101.IUKD011A PART (n/a).
DSNI024I - DSNIARPL BACKOUT PROCESSING HAS COMPLETED
        FOR PAGESET DBKD0101.TLKD0101 PART (n/a).
DSN9022I - DSNVRP 'RECOVER POSTPONED' NORMAL COMPLETION
```

If no postponed units of recovery are found, the following output is returned:

```
DSNV434I - DSNVRP NO POSTPONED ABORT THREADS FOUND
DSN9022I - DSNVRP 'RECOVER POSTPONED' NORMAL COMPLETION
```

Chapter 59. -RESET GENERICLU (DB2)

The RESET GENERICLU command allows you to purge information stored by VTAM in the coupling facility for one or more partners of a particular DB2 subsystem. The command must be issued from the DB2 subsystem that has the VTAM affinity to the particular partner LU whose information you are purging.

Abbreviation: -RESET GENERIC

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions" on page 322
- "Usage notes" on page 322
- "Examples" on page 322

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

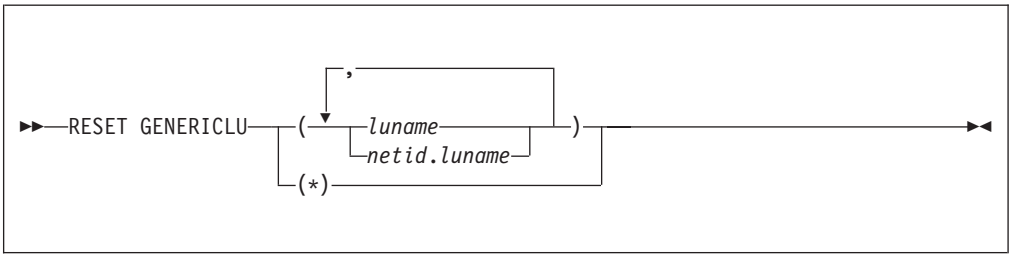
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



© Copyright IBM Corp. 1983, 2008

Option descriptions

(*luname***)**
Specifies the real VTAM LU name of the partner whose generic LU name mapping is to be purged. The NETID of this partner LU must be the same as the local DB2 NETID.

(*netid.luname***)**
Specifies that the VTAM shared memory information that is associated with the specified NETID and LUNAME is purged.

(*) Purges the VTAM shared memory information for all partners of this DB2 subsystem. This command option should only be used if you are planning to remove this DB2 subsystem from the DB2 group.

Usage notes

The following conditions must be satisfied for the RESET GENERICLU command to be successful:

- DDF must be started.
- No VTAM sessions can be active to the partner LU that is specified on the command.
- DB2 must not have any indoubt thread resolution information associated with the specified partner LU.

Examples

Example 1: Purge the VTAM generic name mapping that is associated with partner NET1.USER5LU.

```
-DB2A RESET GENERICLU(NET1.USER5LU)
```

Example 2: Purge the VTAM generic name mappings for all LUs that are partners of this DB2 subsystem. Use this version of the command only when removing this DB2 subsystem from the data sharing group.

```
-DB2A RESET GENERICLU(*)
```

Chapter 60. -RESET INDOUBT (DB2)

The DB2 command RESET INDOUBT purges the information that is displayed in the indoubt thread report that is generated by the DISPLAY THREAD command.

This command **must** be used to purge indoubt thread information in the following situations:

- For threads where DB2 has a coordinator responsibility that it cannot fulfill because of participant cold start, sync point protocol errors, or indoubt resolution protocol errors.
- For threads that were indoubt but were resolved with the RECOVER INDOUBT command, and subsequent resynchronization with the coordinator shows heuristic damage.

  The RESET column of a display thread report for indoubt threads indicates whether information in the report must be purged with this command.

This command can also be used to purge indoubt thread information for threads where:

- DB2 has a coordinator responsibility even when no errors have been detected that preclude automatic resolution with the participants. The FORCE keyword must be specified to purge this information. Resynchronization with affected participants is not performed.
- DB2 has a participant responsibility even when no errors have been detected that preclude automatic resolution with the coordinator. Resynchronization with the coordinator will not be performed.

Abbreviation: -RESET IND

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 324
- "Option descriptions" on page 324
- "Output" on page 325
- "Usage notes" on page 325

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

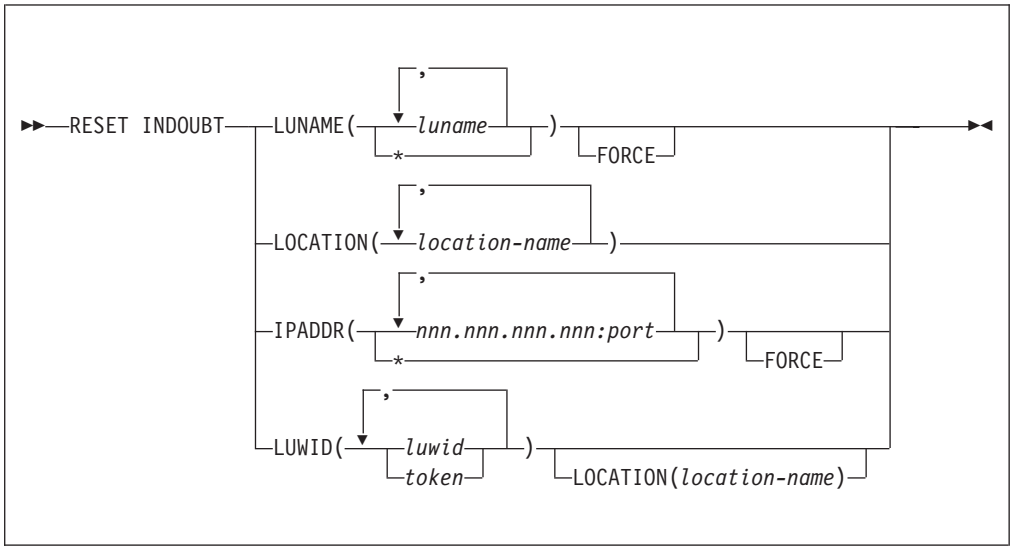
- RECOVER privilege
- SYSOPR authority

© Copyright IBM Corp. 1983, 2008

- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Option descriptions

LUNAME(*luname***, ...)**
Purges all qualifying indoubt information that pertains to the specified LUNAME.

luname
Is expressed as a one- to eight-character name. If you use more than one LUNAME, separate each name with a comma.

(*) Purges indoubt information for all SNA locations.

FORCE
Forces the purging of coordinator and participant indoubt resolution responsibility even when no errors that preclude automatic resolution have been detected. FORCE can be used in conjunction with IPADDR or LUNAME.

Purging resynchronization information when no errors that preclude automatic resynchronization have been detected simulates a cold start. Thus, no connections can exist between DB2 and the named partner when this command is executed. After you run the FORCE option, the next connection with the named partner location will be a cold start connection. If a connection with the named partner exists at the time this command is run, the command fails with message DSNL448I.

FORCE can be used to bypass warm start connectivity problems when errors that are occurring in the recovery log name exchange result in the partner refusing the connection attempt.

LOCATION(*location-name***, ...)**
Purges all qualifying indoubt information that pertains to the named location.

location-name is expressed as a 1- to 16-character name, which identifies the partner, whether it is a requester or server. If the partner is not a DB2 UDB for z/OS subsystem, the location name can be expressed as one of the following formats:

- A one- to eight-character luname, as defined to VTAM at the server location. This name must be enclosed in the less-than (<) and the greater-than (>) characters to distinguish it from a DB2 location name.
- A dotted decimal TCP/IP address.

IPADDR(*nnn.nnn.nnn.nnn:port***)**
Purges all qualifying indoubt information that pertains to the dotted decimal IP address that is associated with the resync port number.

This keyword can be used in place of the LUNAME keyword when the partner uses TCP/IP instead of SNA.

nnn.nnn.nnn.nnn:port
Is the dotted decimal IP address of the remote site, followed by the resync port number. If you use more than one IP address and port number, use commas to separate the items in the list.

(*) Purges indoubt information for all TCP/IP locations.

LUWID
Purges indoubt information for the thread with the specified LUWID.

luwid
Consists of an LU network name, an LUW instance number, and a commit sequence number.

The LU network name consists of a 1- to 8-character network ID, a period, and a 1- to 8-character network LU name. The LUW instance number consists of a period followed by 12 hexadecimal characters. The last element of the LUWID is the commit sequence number, which consists of a period followed by four hexadecimal characters.

token
A token is an alternate way to express an LUWID. DB2 assigns a token to each thread that it creates. It is a one- to six-digit decimal number that appears after the equal sign in all DB2 messages that display an LUWID.

If you enter one- to six-decimal digits, DB2 assumes that you are supplying a token. The token that DB2 assigns to a specific LUWID is unique for that DB2 subsystem, but it is not necessarily unique across all subsystems.

Output

The response from this command includes any of the messages from DSNL440I through DSNL449I.

If you specify RESET INDOUBT incorrectly, you receive message DSNL440I.

Usage notes

Purging participant indoubt information: Use caution when you specify the FORCE option to purge participant indoubt information. Normally, after the use of the RECOVER INDOUBT command, automatic resolution with the coordinator determines if heuristic damage has occurred. This detection is lost if RESET INDOUBT is used before automatic resolution with the coordinator can be achieved.

Purging coordinator indoubt information: Use caution when you specify the FORCE option to purge coordinator indoubt information when no errors are precluding automatic resolution. When the information is purged, any participant that is indoubt is forced to use a heuristic decision process to resolve the indoubt logical unit of work.

Chapter 61. RUN (DSN)

The DSN subcommand RUN executes an application program, which can contain SQL statements.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage note" on page 329
- "Examples" on page 329

Environment

This subcommand can be issued under the DSN command processor running in either foreground or background mode, or it can be issued by using the DB2I RUN panel.

Data sharing scope: Member

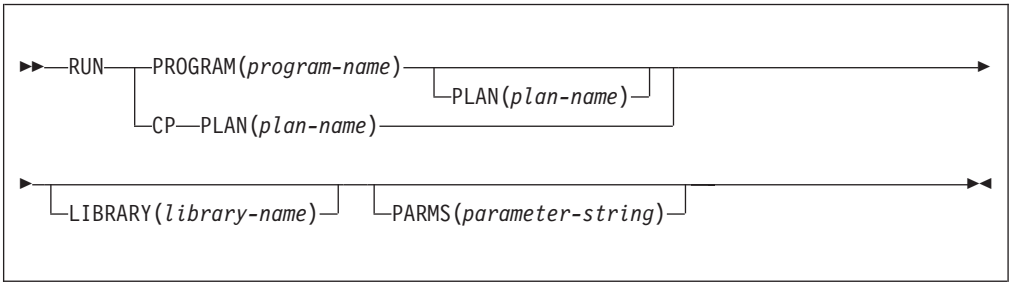
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- EXECUTE privilege on the plan
- Ownership of the plan
- SYSADM authority

To run an application, the plan must be enabled for your local server. Any associated packages from which you execute statements must also be enabled.

Syntax



Option descriptions

Use at least one of the two following clauses, but do not use the same clause twice.

PROGRAM (*program-name***)**
Identifies the program that you want to run.

© Copyright IBM Corp. 1983, 2008

CP
Directs input to the user's command processor, and causes a prompt to be issued: `ENTER TSO COMMAND`. This is useful for running command processors and debugging programs (for example, COBTEST).

Processing the specified TSO command creates a new task control structure under which the TSO command executes. All application programs that are initiated from this TSO command session also execute under the same task structure, and must establish a new connection to DB2 if they use SQL requests.

When the TSO command completes, the new task structure is terminated, and control is returned to the original DB2 connection and task structure established by the DSN command.

Later TSO commands can be issued directly from the DSN session, or through the RUN subcommand with the CP option.

PLAN(*plan-name***)**
Is optional after the PROGRAM option, but required after the CP option.

plan-name is the name of the application plan for the program.

When PROGRAM is used, the **default** plan name is *program-name*.

LIBRARY(*library-name***)**
Specifies the name of the data set that contains the program to be run.

If *library-name* is not specified, normal z/OS library searching is used. The data sets that are specified in the STEPLIB DD statements are first searched for the entry point name of the program. If STEPLIB is not present, the data sets that are specified in the JOBLIB DD statements are searched. If the entry point name is not found there, the link list is searched.

Subprograms: Normal z/OS library searching is **always** used for any subprograms that is loaded by the main program. If the subprograms reside in the same library as the main program, the library-name must also be defined for the normal z/OS search pattern (STEPLIB, JOBLIB, link list). If a library that is defined in that way contains both the main program and any loaded subprograms, you do not need to use the LIBRARY option.

PARMS(*parameter-string***)**
parameter-string is a list of parameters that are to be passed to your application program. Separate items in the list with commas, blanks, or both, and enclose the list between apostrophes. If the list contains apostrophes, represent each apostrophe by using two consecutive apostrophes. The list is passed as a varying-length character string of 1- to 100-decimal characters.

For Assembler: Use a list of the form 'program parameters'. There are no run-time parameters.

No run-time or application parameter validation is performed by the RUN subcommand on the *parameter-string* that is passed to your application program. All specified parameter values are assumed to adhere to the parameter syntax and format criteria defined by the language in which the application program is written.

For C: Use a list of the form A/B, where A represents a list of run-time options, and B represents a list of parameters for the C application program. If run-time options are not needed, write the list in the form /B. If the NOEXECOPS run-time option is in effect, omit the "/".

For COBOL: If Language Environment is not the run-time environment, use a list of the form B/A, where B represents a list of parameters for the COBOL application program, and A represents a list of run-time options. If program parameters are not needed, write the list in the form of /A.

If Language Environment is the run-time environment, use a list of the form A/B, where A represents a list of run-time options, and B represents a list of parameters for the COBOL application program. If run-time options are not needed, write the list in the form of /B. For compatibility, Language Environment provides the CBLOPTS run-time option. When CBLOPT(YES) is specified in CEEDOPT or CEEUOPT and the main routine is COBOL, specify the list in the form of B/A, the same form as when the run-time environment is not Language Environment. CBLOPT(NO) is the default.

For Fortran: Use a list of the form A/B, where A represents a list of Fortran run-time options and B represents a list of parameters for the Fortran application program. If Fortran run-time options are not needed, write the list in the form of B or /B. The second form must be used if a slash is present within the program arguments. If only Fortran run-time options are present, write the list in the form of A/.

For PL/I: Use a list of the form A/B, where A represents a list of run-time options, and B represents a list of parameters for the PL/I application program. If run-time options are not needed, write the list in the form /B. If the PL/I NOEXECOPS procedure option is specified, omit the "/". An informational system message is issued if you omit the slash, or if the value that is passed to the PL/I run-time package is not valid.

Usage note

Multitasking restriction: When running a program that uses a multitasking environment, the first task to issue an SQL statement must issue all subsequent SQL calls. That is, only one task in a multitasking environment can issue SQL calls. This task must be a subtask of, or running at the same TCB level as, the DSN main program.

Examples

Example 1: Run application program DSN8BC4. The application plan has the same name. The program is in library '*prefix*.RUNLIB.LOAD'.

```
DSN SYSTEM (DSN)
RUN PROGRAM (DSN8BC4) LIB ('prefix.RUNLIB.LOAD')
```

Example 2: Run application program DSN8BP4. The application plan is DSN8BE81. The program is in library '*prefix*.RUNLIB.LOAD'. Pass the parameter O'TOOLE to the PL/I application program with no PL/I run-time options.

```
DSN SYSTEM (DSN)
RUN PROGRAM (DSN8BP4) PLAN (DSN8BE81) -
    LIB ('prefix.RUNLIB.LOAD') PARMS ('/O''TOOLE')
```

Chapter 62. -SET ARCHIVE (DB2)

The DB2 command SET ARCHIVE sets the maximum number of tape units for the archive log. It also sets the maximum deallocation time of tape units for the archive log. This command overrides the values that are specified during installation or in a previous invocation of the SET ARCHIVE command. The changes that SET ARCHIVE makes are temporary; at restart, DB2 again uses the values that are set during installation.

Abbreviation: -SET ARC

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 332
- "Option descriptions" on page 332
- "Usage notes" on page 333
- "Output" on page 333
- "Examples" on page 333

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

Authorization

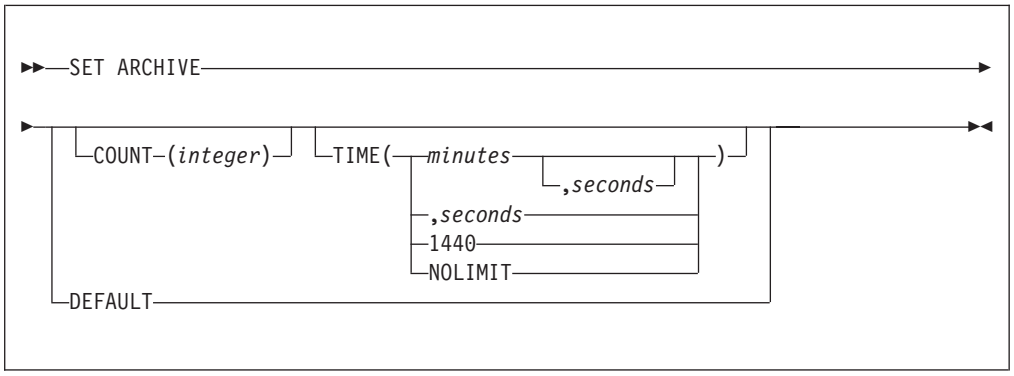
To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- ARCHIVE privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

The following options override the READ TAPE UNITS(COUNT) and DEALLC PERIOD TIME subsystem parameters that are specified during installation.

COUNT(*integer***)**
Specifies the maximum number of tape units that can be dedicated to reading archive logs. This value affects the number of concurrent reads that are allowed for unique archive data sets that reside on tapes.

integer can range from 1 to 99.

- If the number that you specify is greater than the current specification, the maximum number of tape units allowable for reading archive logs increases.
- If the number that you specify is less than the current specification, tape units that are not being used are immediately deallocated to adjust to the new COUNT value. Active (or premounted) tape units remain allocated; only tape units that are inactive are candidates for deallocation because of a lowered COUNT value.

TIME
Specifies the length of time during which an allocated archive read tape unit is allowed to remain unused before it is deallocated.

(*minutes***)**
Specifies the maximum number of minutes.

minutes must be an integer between 0 and 1439.

(*seconds***)**
Specifies the maximum number of seconds.

seconds must be an integer between 1 and 59.

(NOLIMIT) or (1440)
Indicates that the tape unit will never be deallocated. Specifying TIME(1440) is equivalent to TIME(NOLIMIT). The seconds specification is not allowed when you specify that TIME is 1440.

DEFAULT
Resets the COUNT and TIME parameters back to the values that were specified during DB2 installation.

Usage notes

Archive tape reading performance: To achieve the best performance for reading archive tapes, specify the maximum values that are allowed (within system constraints) for both the COUNT and TIME options.

IEF238D "REPLY DEVICE NAME OR CANCEL" message: Replying "CANCEL" to this message resets the COUNT value to the current number of tape units. For example, if the current COUNT value is 10, but you reply "CANCEL" to the request for the seventh tape unit, the COUNT value is reset to 6.

Delaying tape deallocation in a data sharing environment: When you submit a recover job on a member of a data sharing group that requires a tape unit that must remain unused for a certain length of time before it is deallocated, the archive tape is not available to any other member of the group until the tape is deallocated. Unless all recover jobs will be submitted from the same member, you might not want to use the COUNT option and ensure that field DEALLOC PERIOD on installation panel DSNTIPA has a value of 0.

Output

The response from this command includes any of the messages from DSNJ334I through DSNJ337I.

Examples

Example 1: Allocate two tape units that can remain unused for 30 seconds before they are deallocated.

```
-SET ARCHIVE COUNT(2) TIME(,30)
```

Example 2: Allocate four tape units that can remain unused for 2 minutes before they are deallocated.

```
-SET ARCHIVE COUNT(4) TIME(2)
```

Example 3: Allocate one tape unit that is never deallocated.

```
-SET ARCHIVE COUNT(1) TIME(1440)
```

Chapter 63. -SET LOG (DB2)

The DB2 command SET LOG modifies the checkpoint frequency that is specified during installation. This command also overrides the value that was specified in a previous invocation of the SET LOG command. The changes that SET LOG makes are temporary; at restart, DB2 again uses the values that were set during installation. The LOGLOAD value takes effect following the next system checkpoint. SET LOG can also be used to SUSPEND or RESUME logging and update activity for the current DB2 subsystem.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions" on page 336
- "Usage notes" on page 337
- "Examples" on page 337

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program that uses the instrumentation facility interface (IFI).

Data sharing scope: Member

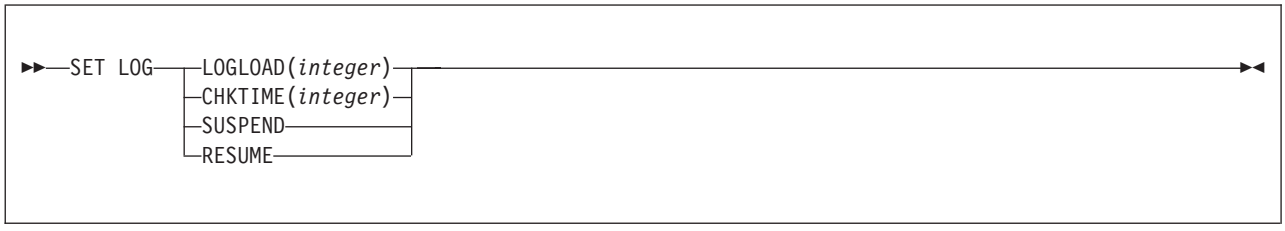
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- ARCHIVE privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



© Copyright IBM Corp. 1983, 2008

Option descriptions

The following option overrides the LOGLOAD subsystem parameter that is specified in the CHECKPOINT FREQ field on installation panel DSNTIPN.

LOGLOAD(*integer*)
Specifies the number of log records that DB2 writes between the start of successive checkpoints. You can specify a value of 0 to initiate a system checkpoint without modifying the current LOGLOAD value.

integer can be 0, or within the range from 200 to 16000000.

CHKTIME(*integer*)
Specifies the number of minutes between the start of successive checkpoints. This option overrides log records that are specified by installation options or the LOGLOAD option that are based on checkpoint frequency.

integer can be any integer value from 0 to 60. Specifying 0 starts a system checkpoint immediately without modifying the checkpoint frequency.

SUSPEND
Specify to suspend logging and update activity for the current DB2 subsystem until SET LOG RESUME is issued. DB2 externalizes unwritten log buffers, takes a system checkpoint (in non-data-sharing environments), updates the BSDS with the high-written RBA, and then suspends the update activity. Message DSNJ372I is issued and remains on the console until update activity resumes.

SUSPEND quiesces the writes for 32-KB pages and the data set extensions for all page sizes. If a 32-KB page write is in progress when you take volume-level copies of your data, SUSPEND prevents an inconsistent copy of a 32-KB page when the copy of your data is restored. If a data set extension is in progress, SUSPEND prevents inconsistencies between the VSAM catalog and the DB2 data set when the copy of your data is restored.

This option is not allowed when the ARCHIVE LOG or STOP DB2 commands activate a system quiesce. Update activity remains suspended until SET LOG RESUME or STOP DB2 is issued. (Also, when logging is suspended, do not issue the ARCHIVE LOG command without also specifying CANCEL OFFLOAD.)

Recommendation: Do not keep log activity suspended during periods of high activity or for long periods of time. Suspending update activity can cause timing-related events such as lock timeouts or DB2 and IRLM diagnostic dumps.

RESUME
Specify to resume logging and update activity for the current DB2 subsystem and to remove the message DSNJ372I from the console. Resumes 32-KB page writes and data set extensions for pages of all sizes.

Recommendation: Issue this command from a z/OS console or from the installation SYSADM ID to avoid possible contention during command authorization checking. When logging is suspended by the SET LOG SUSPEND command, the contention that is generated by holding the log-write latch can cause command-authorization checking to hang until logging resumes.

Usage notes

How LOGLOAD and CHKTIME values affect DB2 performance: LOGLOAD and CHKTIME values can affect the amount of time needed to restart DB2 after abnormal termination. A large value for either option can result in lengthy restart times. A low value can result in DB2 taking excessive checkpoints. However, when you specify LOGLOAD(0) or CHKTIME(0), the checkpoint request is synchronous when issued from a batch job, and it is asynchronous when issued from a z/OS or TSO console.

The behavior of LOGLOAD(0) and CHKTIME(0) is different in a data sharing
environment. Avoid issuing SET LOG LOGLOAD(0) or SET LOG CHKTIME(0) in
the data sharing environment if logging has been suspended with SET LOG
SUSPEND on any member in the group. If you specify LOGLOAD(0) or
CHKTIME(0), the synchronous checkpoint might be suspended until all logging
has been resumed when you issue the SET LOG RESUME command.

Use the DISPLAY LOG command to display the current LOGLOAD setting.

The value that you specify for LOGLOAD or CHKTIME is reset to the value specified in the subsystem parameter when DB2 is restarted. If you load a different value by issuing the command SET SYSPARM, the new value is used.

When to suspend logging: Specify SET LOG SUSPEND before making a remote copy of the entire database and logs for a system-level, point-in-time recovery or disaster recovery. You can make remote copies with peer-to-peer remote recovery (PPRC) and FlashCopy®. Suspending logging to make a remote copy of the database lets you avoid quiescing update activity. Read-only activity continues while logging is suspended.

The backup that is made between the SET LOG SUSPEND and the SET LOG RESUME window might contain uncommitted data. If you must restore the entire DB2 subsystem to the time when the log was suspended, restore the entire database and logs from the backup, and then restart DB2 to recover the entire DB2 subsystem to a consistent state. For details, see Part 4 (Volume 1) of *DB2 Administration Guide*.

Examples

Example 1: Initiate a system checkpoint without modifying the current LOGLOAD value.

```
-SET LOG LOGLOAD(0)
```

Example 2: Modify the system checkpoint interval to every 150000 log records.

```
-SET LOG LOGLOAD(150000)
```

Example 3: Suspend logging activity.

```
-SET LOG SUSPEND
```

Example 4: Resume logging activity.

```
-SET LOG RESUME
```

Chapter 64. -SET SYSPARM (DB2)

The DB2 command SET SYSPARM lets you change subsystem parameters while DB2 is up.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage notes" on page 340
- "Examples" on page 340

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program that uses the instrumentation facility interface (IFI).

Data sharing scope: Member

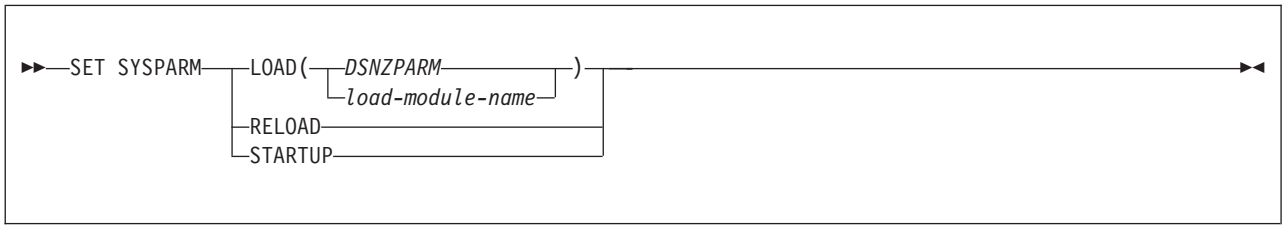
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Option descriptions

LOAD(*load-module-name***)**
Specifies the name of the load module to load into storage. The default load module is **DSNZPARM**.

© Copyright IBM Corp. 1983, 2008

RELOAD
Reloads the last named subsystem parameter load module into storage.

STARTUP
Resets loaded parameters to their startup values.

Usage notes

To update the subsystem parameters on a subsystem, follow these steps:

1. Run through the installation process in Update mode.
2. Produce a new subsystem parameter load module.
3. Issue the SET SYSPARM command.

You must use an authorization that includes installation SYSADM authority to change the installation SYSADM (SYSADM1 and SYSADM2) subsystem parameters. You must also use an authorization that includes installation SYSADM authority to change the installation SYSOPR (SYSOPR1 and SYSOPR2) subsystem parameters.

If you attempt to change installation SYSADM or installation SYSOPR subsystem parameters and you do not have the proper authority, the parameter values that are in place prior to the load of the new subsystem-parameter module are used instead of the unauthorized values in the new module. DB2 issues message DSNZ015 for each attempt of an unauthorized change to a subsystem parameter.

Examples

Example 1: Change from DSNZPARM to ADMPARM1.

```
-SET SYSPARM LOAD(ADMPARM1)
```

Example 2: Reload ADMPARM1 if it is the currently running load module.

```
-SET SYSPARM RELOAD
```

Example 3: Reload the subsystem parameters that the DB2 subsystem loaded at startup.

```
-SET SYSPARM STARTUP
```

Chapter 65. SPUFI (DSN)

The DSN subcommand SPUFI executes the SQL processor using file input.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Usage notes"

Environment

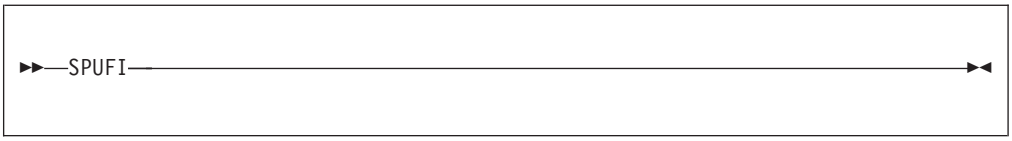
You can use this subcommand only under ISPF. You can issue it from ISPF option 6, or from a CLIST.

Data sharing scope: Member

Authorization

None is required.

Syntax



Usage notes

SPUFI session: The SPUFI subcommand runs SPUFI and presents the SPUFI panel as the start of a SPUFI session. For a description of the panel and instructions on using SPUFI, see Part 1 of *DB2 Application Programming and SQL Guide*.

In the SPUFI session, you can access the CURRENT SPUFI DEFAULTS panel. You can change DB2I defaults by splitting the screen and accessing the DB2I DEFAULTS panel, or by changing the defaults before starting the SPUFI session.

SPUFI panel variables: The SPUFI panel variables you enter after invoking SPUFI directly with the DSN command are not saved in the same place. Panel variables, therefore, vary depending on whether you execute the facility directly, or through DB2I.

© Copyright IBM Corp. 1983, 2008

Chapter 66. /SSR (IMS)

The IMS /SSR command allows the IMS operator to enter an external subsystem command.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option description"
- "Usage note"

Environment

This command can be issued only from an IMS terminal.

Data sharing scope: Member

Authorization

This command requires an appropriate level of IMS authority, as described in the *IMS Administration Guide: System*.

In addition, the set of privileges held by the primary authorization ID or any of the secondary authorization IDs must include the authority to enter the DB2 command that follows /SSR. For a description of the privileges required to issue a DB2 command, see the description of the appropriate DB2 command in this book.

Syntax

Option description

subsystem-command
Specifies a valid subsystem command. The first character following /SSR must be the subsystem recognition character of the subsystem to which the command is to be directed (in this case DB2). The IMS subsystem recognition character is defined in the IMS SSM member for the external subsystem.

Usage note

Routing the command: IMS uses the command recognition character (CRC) to determine which external subsystem, in this case DB2, receives the command. The only action taken by IMS is to route the command to the appropriate subsystem.

© Copyright IBM Corp. 1983, 2008

Chapter 67. START admtproc (z/OS)

The START *admtproc*command starts the administrative scheduler that is specified in the *admtproc* parameter.

This command can be started at the operator's console, or during DB2 startup or initialization. After the administrative scheduler is started, it is always on unless a STOP command is issued at the operator's console.

Each DB2 subsystem has a coordinated administrative scheduler address space for starting a z/OS task procedure. Therefore, if more than one DB2 subsystem runs on a single z/OS operating system, each subsystem has an administrative scheduler with a different name. Two instances of the same administrative scheduler cannot run simultaneously. To avoid starting a duplicate administrative scheduler, the administrative scheduler checks all of the address spaces for duplicate names at startup. If another address space with the same name is already running, the duplicate administrative scheduler immediately shuts down and a console error message is issued. The administrative scheduler can only check address spaces within the same system; it cannot check an entire Sysplex.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions" on page 346
- "Example" on page 346

Environment

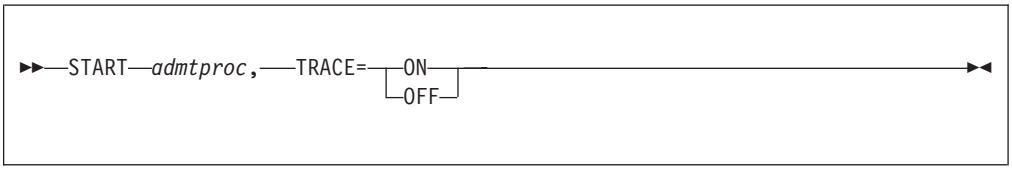
This command can be issued only from a z/OS console.

Data sharing scope: Member

Authorization

The command requires an appropriate level of operating system authority.

Syntax



Options must be separated by commas, with no spaces.

© Copyright IBM Corp. 1983, 2008

Option descriptions

admtproc
Specifies the procedure name of the administrative scheduler task that you want to start.

TRACE=
Specifies whether tracing for the administrative scheduler is to be started (TRACE=ON).

Example

Example: This command starts the *admtproc* scheduler.

Enter the following command on the system console:

```
start admtproc
```

Chapter 68. STOP admtproc (z/OS)

The STOP *admtproc*command stops the administrative scheduler that is specified in the *admtproc* parameter.

This command should only be used to bring down the administrative scheduler for maintenance or to prepare for an IPL. To stop the administrative scheduler for other purposes, issue the command `modify admtproc,appl=shutdown`.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Example"

Environment

This command can be issued only from a z/OS console.

Data sharing scope: Member

Authorization

The command requires an appropriate level of operating system authority.

Syntax

Option descriptions

admtproc
Specifies the procedure name of the administrative scheduler task that you want to stop.

Example

Example: This command stops the *admtproc* scheduler.

Enter the following command on the system console:

```
stop admtproc
```

© Copyright IBM Corp. 1983, 2008

Chapter 69. /START (IMS)

The IMS /START command (with the SUBSYS parameter) makes the connection between IMS and the specified external subsystem available. Establishing the connection allows application programs to access resources managed by the external subsystem.

The following information is only a partial description of the /START command. For a complete description, see *IMS Command Reference.*

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage note" on page 350

Environment

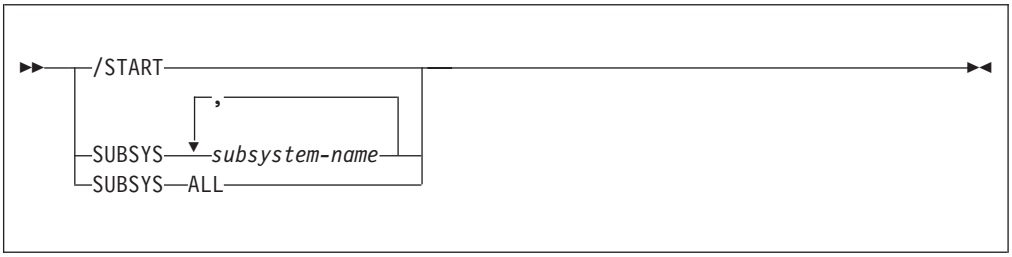
This command can be issued only from an IMS terminal.

Data sharing scope: Member

Authorization

This command requires an appropriate level of IMS authority, as described in the *IMS Administration Guide: System.*

Syntax



Option descriptions

SUBSYS
Specifies one or more names of external subsystems to be connected to IMS, or all external subsystems.

subsystem-name, **...**
Identifies one or more names of external subsystems to be connected to IMS.

ALL
Indicates that all external subsystems are to be connected to IMS.

© Copyright IBM Corp. 1983, 2008

Usage note

Inactive entries: The copy in main storage of the external subsystem PROCLIB entry is refreshed as part of /START command function when that entry is not active (that is, when the connection does not exist). This allows the installation to stop the subsystem connection, change the specifications in the PROCLIB entry, and restart the subsystem connection without bringing down IMS.

Chapter 70. -START DATABASE (DB2)

The START DATABASE command makes the specified database available for use. Depending on which options you specify, the following objects can be made available for read-only processing, read-write processing, or utility-only processing.

- Databases
- Table spaces
- Index spaces
- Physical partitions of partitioned table spaces or index spaces (including index spaces housing data-partitioned secondary indexes (DPSIs))
- Logical partitions of nonpartitioned secondary indexes.

The command is typically used after one of the following events:

- The STOP DATABASE command is issued
- A table space, partition, or index is placed in group buffer pool RECOVER-pending status (GRECP)
- Pages have been put on the logical page list (LPL) for a table space, partition, or index

In a data sharing environment, the command can be issued from any DB2 subsystem in the group that has access to the specified database.

Abbreviation: -STA DB

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 353
- "Option descriptions" on page 353
- "Usage notes" on page 356
- "Examples" on page 358

Environment

This command can be issued from a z/OS console, a DSN session, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group

Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities:

- STARTDB privilege
- DBMAINT authority
- DBCTRL authority
- DBADM authority
- SYSCTRL authority

© Copyright IBM Corp. 1983, 2008

- SYSADM authority

When you are using a privilege set that does not contain the STARTDB privilege for a specified database, DB2 issues an error message.

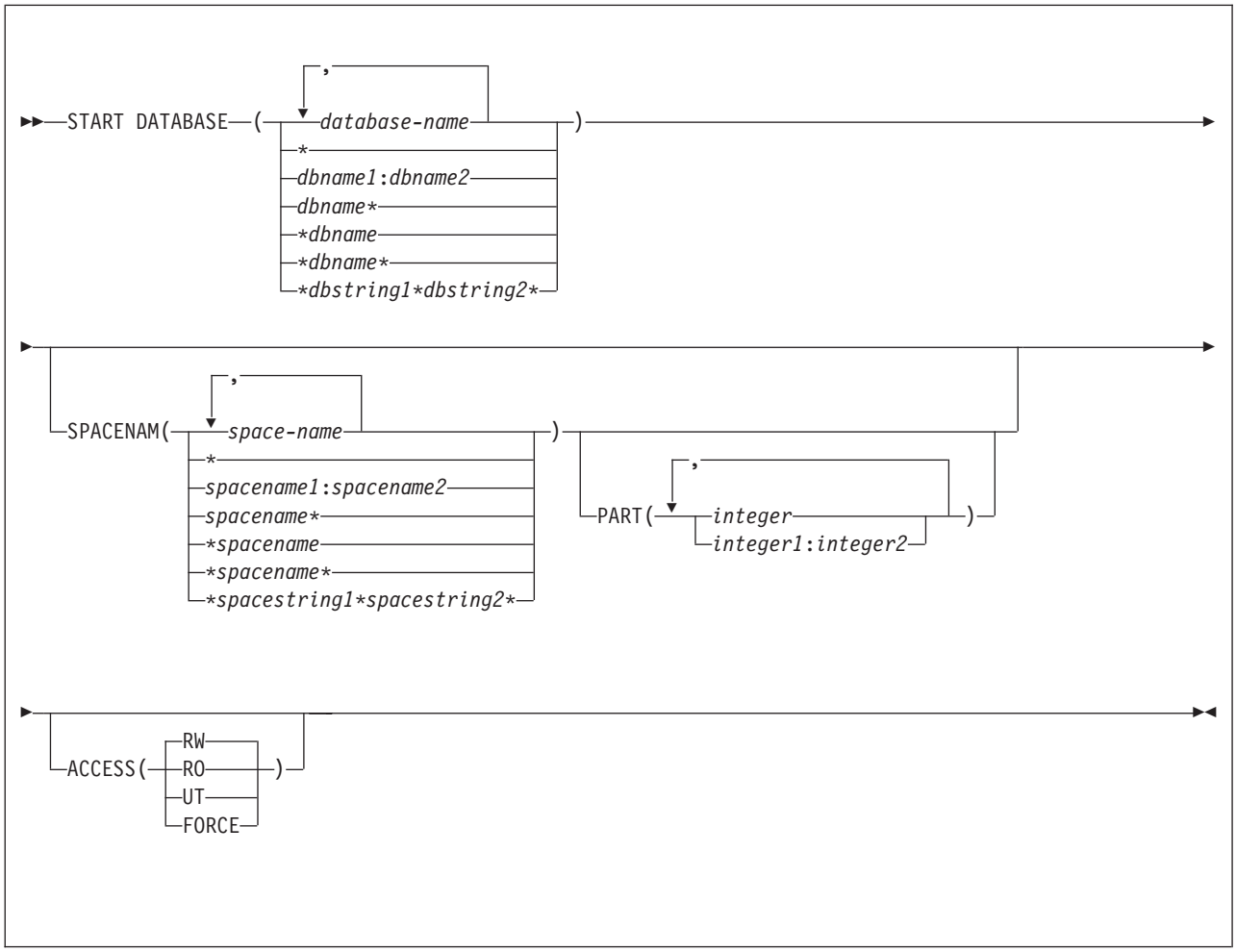
All specified databases with the STARTDB privilege included in the privilege set of the process are started.

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

When data definition control is active, installation SYSOPR or installation SYSADM authority is required to start a database, a table space, or an index space containing a registration table or index.

Table space DBD01 in database DSNDB01 and table spaces and index spaces in database DSNDB06 may be required to check the authorization for using the START DATABASE check the authorization for using the START DATABASE command. If a table or index space required for this authorization check is stopped, or is otherwise unavailable, then installation SYSADM authority is required to start any database, table space, or index space, including the ones required for the authorization check. If a table space or index space is in LPL or GRECP status, installation SYSOPR can recover it using START DATABASE, but might not be allowed to change the access mode.

Syntax



Option descriptions

(*database-name***, ...)**
Specifies the name of a database, or a database for the table spaces or index spaces that are to be started. If you use more than one name, separate names in the list with commas.

(*) Starts all databases for which the privilege set of the process has at least DBMAINT authority or STARTDB privilege (except databases that are already started). You cannot use (*) with ACCESS(FORCE).

You can start DSNDB01, DSNDB06, and work file databases, such as DSNDB07, only by explicitly specifying them (for example, START DATABASE(DSNDB01)).

dbname and *dbstring* can have any of the forms in the following list (where *dbname1* and *dbname2* represent any 1- to 8-character string, and *dbname* represents any 1- to 7-character string):

Form	Starts

dbname1:dbname2	All databases whose names are greater than or equal to *dbname1* and less than or equal to *dbname2*
*dbname**	All databases whose names begin with the string *dbname*
**dbname*	All databases whose names end with the string *dbname*
dbname	All databases whose names contain the string *dbname*
dbstring1*dbstring2	All databases whose names contain the strings *dbstring1* and *dbstring2*

SPACENAM
Specifies the particular table spaces or indexes within the database that are to be started. If you use ACCESS(FORCE), you must use SPACENAM with a list of table space and index names.

Abbreviation: SPACE, SP

(*space-name***, ...)**
Specifies the name of a table space or index space that is to be started. You can use a list of several names of table spaces and index spaces. Separate names in the list with commas.

You can specify *space-name* like *database-name* to designate:
- The name of a single table space or index space
- A range of names
- A partial name, including a beginning or ending pattern-matching character (*)
- Two strings separated by a pattern-matching character (*)
- Any combination of the previous items in this list, with the following exceptions. Consecutive pattern-matching characters (*) are not allowed, and you cannot specify two pattern-matching characters (*) in the middle of a keyword string.

You cannot use a partial name or a range of names with the ACCESS(FORCE) option.

(*) Starts all table spaces and index spaces in the specified database. You cannot use (*) with ACCESS(FORCE).

spacename and *spacestring* can have any of the forms in the following list (where *spacename1* and *spacename2* represent any 1- to 8-character string, and *spacename* represents any 1- to 7-character string):

Form	Displays the status of
spacename1:spacename2	All table spaces or index spaces whose names are greater than or equal to *spacename1* and less than or equal to *spacename2*
*spacename**	All table spaces or index spaces whose names begin with the string *spacename*
**spacename*	All table spaces or index spaces whose names end with the string *spacename*

spacename	All table spaces or index spaces whose names contain the string *spacename*
spacestring1*spacestring2	All table spaces or index spaces whose names contain the strings *spacestring1* and *spacestring2*

PART (*integer***, ...)**
Specifies the partition number of one or more partitions, within the specified table space or index, that are to be started. The start or stop state of other partitions does not change.

The specified *integer* must identify a valid partition number for the corresponding space name and database name. If you specify nonvalid partition numbers, you receive an error message for each nonvalid number, but all other valid partitions that you specified are started.

integer can be written to designate one of the following specifications:

- A list of one or more partitions
- A range of all partition numbers that are greater than or equal to *integer1* and less than or equal to *integer2*
- A combination of lists and ranges

The PART option is valid with partitioned table spaces, partitioned indexes, and nonpartitioned type 2 indexes of partitioned table spaces. If you specify PART with a nonpartitioned table space or index on a nonpartitioned table space, you receive an error message, and the nonpartitioned space is not started.

ACCESS
Specifies whether the objects that are started are in read/write, read only, or utility only status. Also forces access to objects that are in unavailable status.

Abbreviation: ACC

(RW)
Allows programs to read from and write to the specified databases, table spaces, indexes, or partitions.

(RO)
Allows programs to only read from the specified databases, table spaces, indexes, or partitions. Any programs attempting to write to the specified objects will not succeed. Do not use this option for a database for declared temporary tables (databases created with the AS TEMP option).

(UT)
Allows only DB2 online utilities and the SQL DROP statement to access the specified databases, table spaces, indexes, or partitions.

(FORCE)
Resets any indications that a table space, index, or partition is unavailable because of pages in the logical page list, pending-deferred restarts, write-error ranges, read-only accesses, or utility controls. FORCE also resets the CHECK-pending, COPY-pending, and RECOVER-pending states. Full access to the data is forced. FORCE cannot be used to reset the restart-pending (RESTP) state.

When using ACCESS(FORCE), you must use a single database name, the SPACENAM option, and an explicit list of table space and index names. You cannot use any range or combination of pattern-matching characters (*), including DATABASE (*) or SPACENAM (*).

A utility-restrictive state is reset (and the utility is terminated) only if all of the target objects are reset with this command. To identify which objects are target objects of the utility, use the DISPLAY DATABASE command, or run the DIAGNOSE utility with the DISPLAY SYSUTIL option. The DIAGNOSE utility should be used only under the direction of IBM Software Support.

Note: ACCESS(FORCE) will not successfully complete if the object you are trying to force was placed in a utility-read-only (UTRO), utility-read-write (UTRW), or utility-utility (UTUT) state by a utility running in a previous release of DB2. If this situation is encountered, DB2 issues message DSNI041I. To reset the restrictive state, you must terminate the utility using the release of DB2 in which it was started.

A table space or index space that is started with ACCESS(FORCE) might be in an inconsistent state. See "Usage notes" for further instructions.

Usage notes

Data sets offline: Disk packs that contain partitions, table spaces, or indexes, do not necessarily need to be online when a database is started. Packs must, however, be online when partitions, table spaces, or indexes are first referred to. If they are not online, an error in opening occurs.

Table spaces and indexes explicitly stopped: If table spaces and indexes are stopped explicitly (using the STOP DATABASE command with the SPACENAM option), they must be started explicitly. Starting the database does not start table spaces or indexes that have been explicitly stopped.

Effect on objects marked with GRECP or with LPL entries: If a table space, partition, or index is in the group buffer pool RECOVER pending (GRECP) status, or if it has pages in the logical page list (LPL), the START DATABASE command begins recovery of the object. You must specify the SPACENAM option and ACCESS (RW) or (RO).

This recovery operation is performed even if SPACENAM specifies an object that is already started.

If the object is stopped when the command is issued, then the START DATABASE command both starts the object and clears the GRECP or LPL status. If the GRECP or LPL recovery action cannot complete, the object is still started.

If any table space or index space that is required to check command authority is unavailable, Installation SYSADM or Installation SYSOPR authority will be required to issue the START DATABASE command. See "Authorization" on page 351 for more details.

When recovering objects that are in GRECP or LPL status, avoid using pattern-matching characters (*) for both the database name and the space name. Multiple START DATABASE(*dbname*) SPACENAM(*) commands running in parallel should complete faster than one START DATABASE(*) SPACENAM(*) command.

If you use pattern-matching characters (*) for both the database name and space name, you must have DBMAINT authority and ensure that the catalog and directory databases have already been explicitly started in the following order:

```
-START DATABASE(DSNDB01) SPACENAM(*)
```

```
-START DATABASE(DSNDB06) SPACENAM(*)
```

Although not recommended, you can start an object using START DATABASE ACCESS(FORCE). That deletes all LPL and write error page range entries without recovering the pages. It also clears the GRECP status.

When starting a LOB table space defined as LOG NO and either in GRECP or having pages in the LPL, the LOB table space will be placed in the AUXW state and the LOB will be invalidated if DB2 detects that log records required for LPL recovery are missing due to the LOG NO attribute.

Use of ACCESS(FORCE): The ACCESS(FORCE) option is intended to be used when data has been restored to a previous level after an error, by DSN1COPY, or by a program that is not DB2 UDB for z/OS, and the exception states resulting from the error still exist and cannot be reset. When using ACCESS(FORCE), it is up to the user to ensure the consistency of data with respect to DB2. For information about DSN1COPY, see *DB2 Utility Guide and Reference*.

If an application process requests a transaction lock on a table space that is in a restrictive status (RECP) or has a required index in a restrictive status, DB2 acquires the lock. DB2 does not detect the status until the application tries to access the table space or index, when the application receives an error message indicating that the resource is not available (SQLCODE -904). After receiving this message, the application should release the lock, either by committing or rolling back (if the value of the RELEASE option is COMMIT) or by ending (if the value of RELEASE is DEALLOCATE). If you issue the command START DATABASE ACCESS(FORCE) for either the table space or the index space while the lock is in effect, the command fails.

If an object has retained locks (that is, a member of a DB2 data sharing group has failed and the locks it held on the object are retained in the lock structure), START DATABASE ACCESS (FORCE) is not allowed.

START DATABASE ACCESS(FORCE) does not execute if postponed abort or indoubt units of recovery exist. If you attempt to issue the START DATABASE ACCESS(FORCE) command in this situation, the command fails. FORCE cannot be used to reset the restart pending (RESTP) state. See Part 2 of *DB2 Utility Guide and Reference* for information about resetting the RESTP state.

Restricted mode (RO or UT): When a START DATABASE command for a restricted
mode (RO and UT) takes effect depends on whether applications are started after
the START DATABASE command has completed, or whether applications are
executing at the time the command is issued. For applications that are started after
START DATABASE has completed, access restrictions are effective immediately. For
applications that are executing at the time START DATABASE is issued, the access
restrictions take effect when the application is allowed to run to completion.
Whether the application is interrupted by the START DATABASE command
depends on various factors. These factors include the ACCESS mode that is
specified on the START DATABASE command, the type of drain activity, if any, on
the table space or partition, and whether any cursors are being held on the table
space or partition.

Do not start table spaces or index spaces for defined temporary tables with RO or UT access. You can start a temporary file database with UT access to accommodate the REPAIR DBD utility.

If the table space, index, or partition must be accessed in a mode that is incompatible with the ACCESS type currently in effect, DB2 issues a resource-unavailable message.

Communications database or resource limit facility: If the communications database (CDB) or resource limit facility (RLF) is currently being used by any member of the data sharing group, any attempt to start either active database or table space with ACCESS(UT) fails.

Synchronous processing completion: Message DSN9022I indicates that synchronous processing has completed successfully.

Asynchronous processing completion: Recovery of objects in GRECP status or with pages on the LPL is performed asynchronously. Message DSNI022I is issued periodically to give you the progress of the recovery. The starting of databases, table spaces, or indexes (a synchronous task) often completes before the recovery operation starts. Therefore, when DB2 issues message DSN9022I, which indicates that synchronous processing has completed, the recovery of objects might not be complete. Message DSNI006I is issued in response to START DATABASE when the object (table space or index space) identified by TYPE and NAME had group buffer pool RECOVER pending (GRECP) or logical page list (LPL) status, and recovery was triggered. The START DATABASE command does not complete until the asynchronous task of recovery completes.

Message DSNI021I indicates that asynchronous processing for an object has completed. You can issue the command DISPLAY DATABASE to determine whether the recovery operation for all objects is complete. If it is complete, the output from the command shows either a RW or a RO status without LPL or GRECP.

Starting a LOB table space: The START DATABASE command can be used to start LOB table spaces and indexes on auxiliary tables. LOB table spaces are started independently of the base table space with which the LOB table space is associated.

Examples

Example 1: Start table space DSN8S81E in database DSN8D81A. Recover the table space if it is in GRECP status or recover the pages on the LPL if one exists.

```
-START DATABASE (DSN8D81A) SPACENAM (DSN8S81E)
```

Example 2: Start all databases (except DSNDB01, DSNDB06, and work file databases) for which you have authority. Recovery for any objects with GRECP or LPL status is not performed.

```
-START DATABASE (*)
```

Example 3: Start the third and fourth partitions of table space DSN8S81E in database DSN8D81A for read-only access. Recover the partitions if they are in GRECP status or recover the pages on the LPL if one exists.

```
-START DATABASE (DSN8D81A) SPACENAM (DSN8S81E) PART (3,4) ACCESS (RO)
```

Example 4: Start all table spaces that begin with ″T″ and end with the string ″IQUA03″ in database DBIQUA01 for read and write access.

```
-START DATABASE (DBIQUA01) SPACENAM (T*IQUA03) ACCESS (RW)
```

This command produces output that is similar to the following output:

```
DSN9022I - DSNTDDIS 'START DATABASE' NORMAL COMPLETION
```

Chapter 71. -START DB2 (DB2)

The DB2 command START DB2 initializes the DB2 subsystem. When the operation is complete, the DB2 subsystem is active and available to TSO applications and to other subsystems (for example, IMS and CICS).

The effect of restarting the system can be controlled by a conditional restart control record, which you create by using the DSNJU003 (change log inventory) utility. For more details about the effects, see "Usage notes" on page 363 and the description of the DSNJU003 utility in *DB2 Utility Guide and Reference*.

Abbreviation: -STA DB2

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 362
- "Option descriptions" on page 362
- "Usage notes" on page 363
- "Examples" on page 364

Environment

This command can be issued only from a z/OS console. The name of the DB2 subsystem is determined by the command prefix. For example, -START indicates that the DB2 subsystem to be started is the one with '-' as the command prefix.

The command is rejected if the DB2 subsystem is already active. The restart recovery status of DB2 resources is determined from the prior DB2 shutdown status.

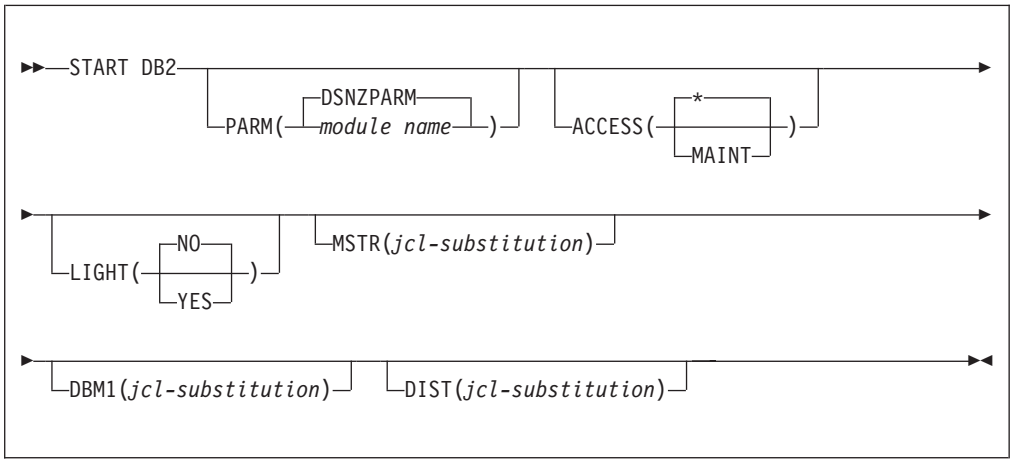
Data sharing scope: Member

Authorization

None is required. However, the command can be executed only from a z/OS console with the START command capability. See *z/OS MVS System Commands*.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

None of the following options are required.

PARM(*module-name***)**
Specifies the load module that contains the DB2 subsystem parameters.

module-name is the name of a load module that is provided by the installation.

The **default** is **DSNZPARM**.

ACCESS
Specifies whether access to DB2 is to be general or restricted.

Abbreviation: ACC

(*) Makes access general; all authorized users can connect to DB2.

The **default** is **ACCESS(*)**.

(MAINT)
Prohibits access to any authorization IDs other than installation SYSADM and installation SYSOPR.

For data sharing, ACCESS(MAINT) restricts access on only the DB2 member on which you execute this command. Other members of the data sharing group are unaffected.

You cannot run stored procedures and user-defined functions when you start DB2 with ACCESS(MAINT).

LIGHT
Specifies whether a light restart is to be performed in a data sharing environment.

(NO)
Restart light is not performed.

(YES)
Specifies that a restart light is to be performed. DB2 starts with reduced storage and terminates normally after freeing retained locks.

You cannot run stored procedures and user-defined functions when DB2 is started with LIGHT(YES).

MSTR(*jcl-substitution***)**
Gives parameters and values to be substituted in the EXEC statement of the JCL that executes the startup procedure for the system services address space.

DBM1(*jcl-substitution***)**
Gives parameters and values to be substituted in the EXEC statement of the JCL that executes the startup procedure for the database services address space.

DIST(*jcl-substitution***)**
Gives parameters and values to be substituted in the EXEC statement of the JCL that executes the startup procedure for the distributed services address space.

(*jcl-substitution***)**
One or more character strings of the form *keyword = value*, enclosed between apostrophes. If you use more than one character string, separate each string with a comma and enclose the entire list between a single pair of apostrophes.

Recommendation: Omit the keyword and use the parameters that are provided in the startup procedure.

Usage notes

Command prefix: If your installation has more than one DB2 subsystem, you must define more than one command prefix.

Conditional restart: A conditional restart control record can prevent a complete restart and specify current status rebuild only. In that case, the following actions occur during restart:

- Log records are processed to the extent that is determined by the conditional restart control record.
- The following values are displayed:
  - The relative byte address (RBA) of the start of the active log
  - The RBA of the checkpoint record
  - The status counts for units of recovery
  - The display table for restart unit of work elements
- The restart operation terminates with an abend.

Light restart with ARM: To enable a light restart in an ARM environment, you must code an ARM policy for DB2 and IRLM.

The following example shows an ARM policy for DB2, where the element name is the DB2 data sharing group name and member name concatenated. For example, DSNDB0GDB1G.

```
ELEMENT(elementname)

   RESTART_METHOD(SYSTERM,STC,'cmdprfx STA DB2,LIGHT(YES)')
```

The following example shows an ARM policy for IRLM, where the element name is the IRLM group name and the ID concatenated. For example, DXRDB0GDJ1G001.

```
ELEMENT(elementname)

   RESTART_METHOD(SYSTERM,STC,'cmdprfx S irlmproc')
```

The element name that DB2 uses is the DB2 data sharing group name and member name concatenated. For example, DSNDB0GDB1G.F

Endless wait during start: The start operation might begin and fail to complete, if the system services address space starts and the database services address space cannot start. If a seemingly endless wait occurs, cancel the system services address space from the console, and check both startup procedures for JCL errors.

Starting members of a data sharing group: To start members of a data sharing group, you must enter a START DB2 command for each subsystem in the group. If it is the first startup of the group, you must start the originating member (the first DB2 that was installed) first.

Examples

Example 1: Start the DB2 subsystem.

```
-START DB2
```

Example 2: Start the DB2 subsystem, and provide a new value for the REGION parameter in the startup procedure for the system services address space.

```
-START DB2 MSTR('REGION=6000K')
```

Example 3: Start the DB2 subsystem. Assuming that the EXEC statement of the JCL that executes the startup procedure for the system services address space uses the symbol RGN, provide a value for that symbol.

```
-START DB2 MSTR('RGN=6000K')
```

Example 4: DB2 subsystems DB1G and DB2G are members of a data sharing group. Both were installed with a command prefix scope of STARTED. Start DB1G and DB2G by routing the appropriate commands to the z/OS system on which they are to be started, MVS1 and MVS2.

```
ROUTE MVS1,-DB1G START DB2
ROUTE MVS2,-DB2G START DB2
```

Chapter 72. -START DDF (DB2)

The DB2 command START DDF starts the distributed data facility (DDF) if it is not already started.

Abbreviation: -STA DDF

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Usage note"
- "Example" on page 366

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

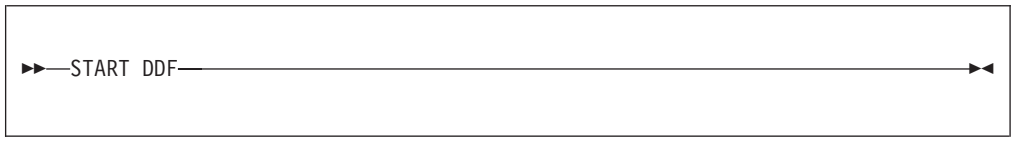
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Usage note

The START DDF command activates the DDF interface to VTAM and TCP/IP. When this command is issued after STOP DDF MODE(SUSPEND), suspended threads are resumed and DDF activity continues.

© Copyright IBM Corp. 1983, 2008

Example

Start the distributed data facility.

```
-START DDF
```

Chapter 73. -START FUNCTION SPECIFIC (DB2)

The DB2 command START FUNCTION SPECIFIC starts an external function that is stopped. Built-in functions or user-defined functions that are sourced on another function cannot be started with this command.

On successful completion of the command, queued requests for the specified functions begin executing. The abend counts for those functions are set to zero.

You do not need to issue the START FUNCTION SPECIFIC command when defining a new function to DB2. DB2 automatically starts the new function on the first SQL statement that invokes the new function.

Historical statistics in the DISPLAY FUNCTION SPECIFIC report (MAXQUE, TIMEOUT) are reset each time a START FUNCTION SPECIFIC command is issued for a given function.

Abbreviation: -STA FUNC SPEC

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 368
- "Option descriptions" on page 368
- "Usage notes" on page 369
- "Examples" on page 369

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel, an IMS or CICS terminal, or a program that uses the instrumentation facility interface (IFI).

Data sharing scope: Group or local, depending on the value of the SCOPE option.

Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities for each function:

- Ownership of the function
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

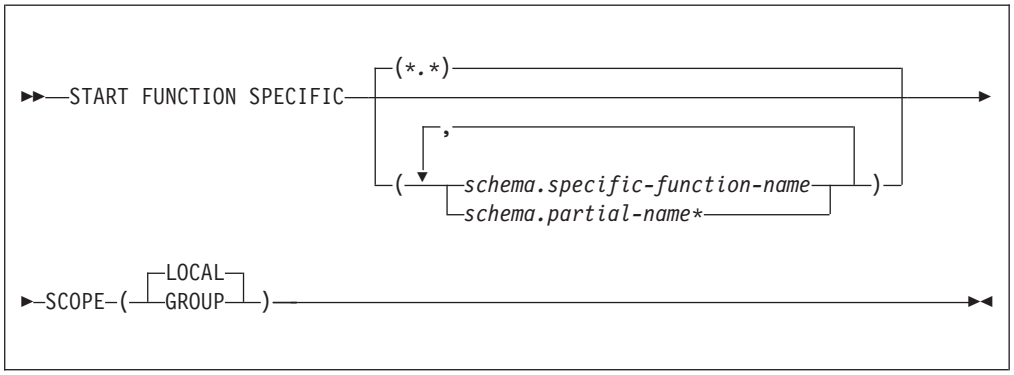
If you specify START FUNCTION SPECIFIC *.* or *schema.partial-name**, the privilege set of the process must include one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

© Copyright IBM Corp. 1983, 2008

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Option descriptions

*** (asterisk)(*.*)**
Starts all functions in all schemas. This is the default.

(*schema.specific-function-name***)**
Starts the specific function name in the schema. You cannot specify a function name in the same way that you do in SQL; you must use the specific name. If a specific name was not specified on the CREATE FUNCTION statement, query SYSIBM.SYSROUTINES for the correct specific name:

```
SELECT SPECIFICNAME, PARM_COUNT
  FROM SYSIBM.SYSROUTINES
   WHERE NAME='function_name'
    AND SCHEMA='schema_name';
```

For overloaded functions, this query can return multiple rows.

(*schema.partial-name****)**
Starts all functions or a set of functions in the specified schema. The specific names of all functions in the set begin with *partial-name* and can end with any string, including the empty string. For example, schema1.ABC* starts all functions with specific names that begin with ABC in schema1.

SCOPE
Specifies the scope of the command.

(LOCAL)
Specifies that the command applies only to the current member.

(GROUP)
Specifies that the command applies to all members of the data sharing group.

Usage notes

Language Environment in the WLM-established stored procedure address space: The START FUNCTION SPECIFIC command does not refresh the Language Environment in the WLM-established stored procedure address space. You must issue the WLM command. For example, if you need to refresh the Language Environment to get new copies of user-defined function load modules, issue the following WLM command:

```
VARY WLM, APPLENV=applenv,REFRESH
```

Examples

Example 1: Start all functions.

```
-START FUNCTION SPECIFIC
```

Output that is similar to the following output is generated:

```
DSN9022I - DSNX9COM '-START FUNC' NORMAL COMPLETION
```

Example 2: Start functions USERFN1 and USERFN2. If any requests are queued for these functions, the functions are executed.

```
-START FUNCTION SPECIFIC(PAYROLL.USERFN1,PAYROLL.USERFN2)
```

Output that is similar to the following output is generated:

```
DSN9022I - DSNX9COM '-START FUNC' NORMAL COMPLETION
```

Chapter 74. START irlmproc (z/OS IRLM)

The START *irlmproc* command starts an IRLM component with a procedure that is defined by the installation. Symbolic parameters in the procedure can be overridden on the START *irlmproc* command.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions" on page 372
- "Examples" on page 375

Environment

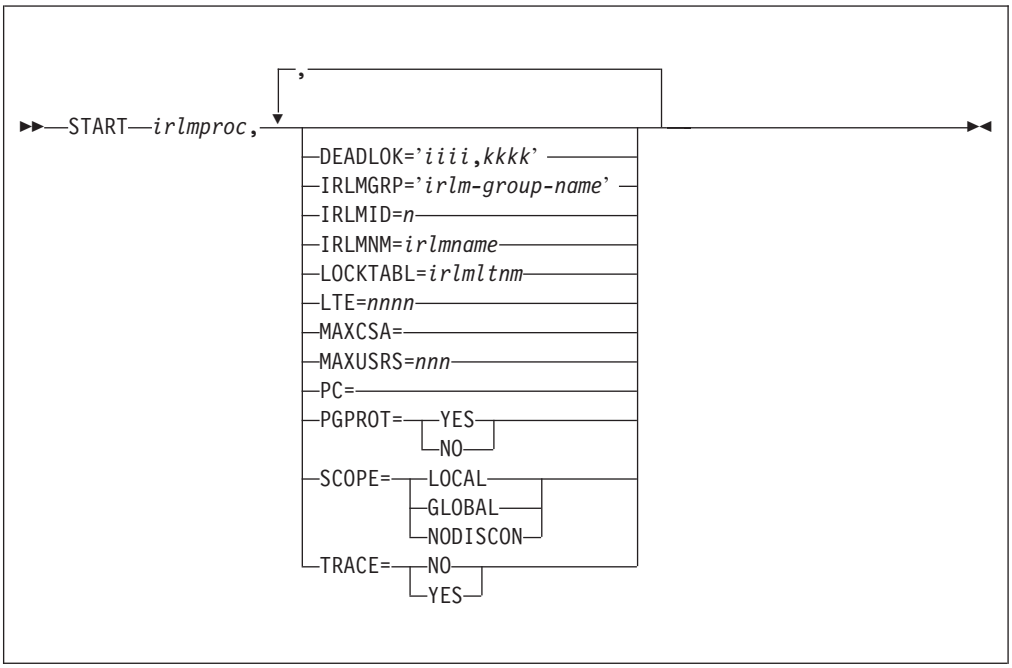
This command can be issued only from a z/OS console.

Data sharing scope: Member

Authorization

The command requires an appropriate level of operating system authority, as described in *z/OS MVS System Commands.*

Syntax



Options must be separated by commas, with no spaces.

© Copyright IBM Corp. 1983, 2008

Option descriptions

irlmproc
Specifies the procedure name of the IRLM to be started.

None of the following options are required:

DEADLOK='*iiii,kkkk***'**
Specifies the local deadlock-detection interval in seconds or milliseconds (*iiii*), and the number of local cycles (*kkkk*) that are to occur before a global detection is initiated.

iiii Is a one- to four-digit number. Values between 1 and 5 are interpreted as seconds. Values between 100 and 5000 are interpreted as milliseconds. Depending on the value that you enter, IRLM might substitute a smaller maximum value.

kkkk
Is a one- to four-digit number from 1 to 9999 that specifies the number of local deadlock cycles that must expire before global deadlock detection is performed. Any value from 1 to 9999 can be specified but IRLM uses 1. The recommended value is 1.

In a data sharing environment, IRLM synchronizes all of the DEADLOK values in the group to the values specified on the most recent IRLM to join the group. The DEADLOK values can be changed by starting a member with the values desired. To reduce confusion, it is recommended that the installation specify the same value for DEADLOK on all of its IRLM start-up procedures and use the START *irlmproc* command to override this value only when the interval must be increased from its original value.

IRLMGRP='*irlm-group-name***'**
Specifies the name of the cross system coupling facility (XCF) group, in a data sharing environment, to which the IRLM belongs as the lock manager for DBMSs that share the same data. All IRLMs in the same group must specify the same value for LOCKTABL and unique values for IRLMID.

The group name is used as the XCF group name. The name must not start with 'SYS' and must not be the same name specified for LOCKTABL.

In a non-data-sharing environment (SCOPE=LOCAL), IRLMGRP is ignored.

IRLMID=*n*
Specifies a decimal number that is used to distinguish between IRLMs in a data sharing group.

n can be either a one- to three-digit number from 1 to 255, or a printable character in quotation marks. Note that this IRLM ID does not relate directly to the limit of IRLM members that can be in the group. That limit is determined by the current hardware limits (currently 32).

When *n* is specified as a printable character, IRLM uses the EBCDIC value of the printable character as the IRLMID (such as X'C4'). The printable character must be surrounded by enough single quotes to permit IRLM to see it as a printable character. Because of the way that the operating system interprets quotes, single quotes must be on either side of the characters. For example, if you want to specify the printable character 'D', you must specify it here as IRLMID='D'.

A unique IRLMID must be specified for each IRLM in a group (IRLMs with the same value specified for the IRLMGRP option).

IRLMNM=*irlmname*
Specifies a 4-byte z/OS subsystem name assigned to this IRLM. (Although z/OS can accept names that are less than 4 bytes, IRLM requires a 4-byte name.)

LOCKTABL=*irlmltnm*
Specifies the lock table to be used by this group. This option is overridden by DB2; it is needed in an IMS environment.

In a non-data-sharing environment (SCOPE=LOCAL), LOCKTABL is ignored.

LTE=*nnnn*
Specifies the number of lock table entries that are required in the coupling facility (CF) lock structure in units of 1048576 entries. LTE= can have a value of blank, zero, or any exact power of two up to 1024 (inclusive). The number of lock table entries in the group is determined by the first IRLM to connect to the group during initial structure allocation or during REBUILD.

The LTE value is used in the following order:

1. The value that is specified using MODIFY irlmproc,SET,LTE= if the value is greater than zero.
2. The value from LTE= in the irlmproc if the value is greater than zero.
3. The value that is determined by the existing logic, which divides the XES structure size returned on the IXCQUERY call by two times LTE width. The result is rounded to the nearest power of two, which the existing logic uses for the value.

   Note: The LTE width is determined by the MAXUSRS value.

If IRLM attempts to use a value from MODIFY irlmproc,SET,LTE= that is greater than the available storage in the structure size returned by XES IXCQUERY, the value for the LTE= in the irlmproc is used. If this value is greater than the available storage, IRLM uses the value that is determined by the existing logic.

Table 30. Some common values for lock table entries and the required lock table storage

For LTE=	Lock Table Storage needed for 2-byte entries	Lock Table Storage needed for 4-byte entries
8	16 MB	32 MB
16	32 MB	64 MB
32	64 MB	128 MB
64	128 MB	256 MB
128	256 MB	512 MB
256	512 MB	1024 MB

MAXCSA=
MAXCSA= is a required positional parameter but is currently unused.

MAXUSRS=*nnn*
Specifies the initial maximum number of members in the data sharing group. IRLM honors the MAXUSRS setting only for the IRLM instance that

causes structure allocation. The specified value determines the size of each lock entry in the lock table portion of the lock structure, as shown in Table 31.

Table 31. Effect of MAXUSRS on initial size of lock table entry

MAXUSRS	Initial size of lock entry
7 or less	2 bytes
≥ 8 and < 24	4 bytes
≥ 24 and < 33	8 bytes

nnn must be a one- to two-digit number from 1 to 32. The default is 7. The recommended value is 7 or less.

In a non-data-sharing environment (SCOPE=LOCAL), MAXUSRS is ignored.

PC=
PC= is a required positional parameter but is currently unused.

PGPROT=
Specifies whether the IRLM load modules that are resident in common storage are placed in z/OS page-protected storage.

YES The IRLM load modules that are resident in common storage are placed in z/OS page-protected storage.

NO The IRLM load modules that are resident in common storage are not placed in z/OS page-protected storage.

SCOPE=
Specifies whether the IRLM is to be used in a data sharing environment.

LOCAL
Specifies the IRLM is in a non-data-sharing environment and there is no intersystem sharing.

GLOBAL
Specifies the IRLM is in a data sharing environment and that intersystem sharing is to be performed.

NODISCON
Specifies that IRLM is in a data sharing environment and that intersystem sharing is to be performed. IRLM remains connected to the data sharing group even when no DBMSs are identified to it. You must explicitly stop IRLM to bring it down.

If you specify the NODISCON option, there is less impact on other systems when a DB2 subsystem fails because the operating system is not required to perform certain recovery actions that it normally performs when IRLM comes down. Using the NODISCON option might allow DB2 to restart more quickly after a DB2 subsystem normally or abnormally terminates because it does not have to wait for IRLM to rejoin the IRLM data sharing group.

TRACE=
Specifies whether the IRLM is to capture traces in wrap-around IRLM buffers. Each buffer is reused when the previous buffer is filled. Traces are captured at IRLM startup. You should specify TRACE=YES in the irlmproc to place traces in wrap-around mode.

NO
Does not capture traces unless the TRACE CT command is issued. See Chapter 89, "TRACE CT (z/OS IRLM)," on page 433 for details.

YES
Captures traces in wrap-around buffers.

Examples

Example: This command starts the IRLM with a lock table storage size of 64 MB, assuming a width of 2-bytes for each lock table entry.

Enter the following command on the system console:

```
S irlmproc,LTE=32
```

If this value is correct, message DXR132I, which is displayed after successful connection to the lock structure, displays the value used by IRLM. If this value is incorrect, START will terminate with DXR116E CODE=24 and ABENDU2018. This value is only used if SCOPE=GLOBAL or SCOPE=NODISCON and has a default value calculated by IRLM.

Chapter 75. -START PROCEDURE (DB2)

For both DB2-established and WLM-established stored procedure address spaces, the DB2 command START PROCEDURE activates the definition of a stored procedure that is stopped. You can qualify stored procedure names with a schema name.

One of the following can also occur:

- If the DB2-established stored procedures address space is not connected to DB2, the operating system starts it.
- If the stored procedures address space is already connected, and some procedure listed in the command is stopped, DB2 stops and restarts the Language Environment environment. At restart, DB2 deletes the existing stored procedure load modules from memory. A deleted load module is reloaded when a CALL statement for that procedure is executed. For WLM-established stored procedures address spaces, a WLM command is needed to do the reload. For example:

  ```
  MVS VARY WLM,APPLENV=applenv,REFRESH
  ```

On successful completion of the command, queued requests for the specified stored procedures begin to execute. The abend counts for the specified procedures are set to zero. DB2 resets the MAXQUE and TIMEOUT statistics to 0 each time that you execute the START PROCEDURE command.

You do not need to issue START PROCEDURE when you define a new stored procedure to DB2. DB2 automatically activates the new definition when it first receives an SQL CALL statement for the new procedure.

Abbreviation: -STA PROC

The following topics provide additional information:

- "Environment"
- "Authorization" on page 378
- "Syntax" on page 378
- "Option descriptions" on page 378
- "Usage notes" on page 379
- "Examples" on page 379

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group or local, depending on the value of the SCOPE option.

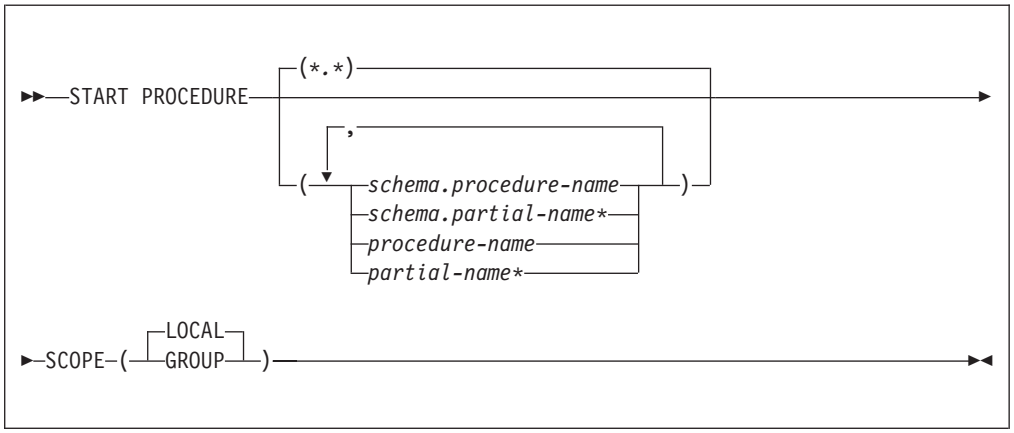
© Copyright IBM Corp. 1983, 2008

Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities:
- Ownership of the stored procedure
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Option descriptions

(*.*)
Marks all stored procedures in all schemas as available to be called.

(*schema.procedure-name***)**
Starts the specified stored procedure in the specified schema.

(*schema.partial-name****)**
Starts a set of stored procedures in the specified schema. The names of all procedures in the set begin with *partial-name* and can end with any string, including the empty string. For example, PAYROLL.ABC* starts all stored procedures with names that begin with ABC in the PAYROLL schema.

procedure-name
Marks one or more specific stored procedures as available to be called.

*partial-name**
Marks a set of stored procedures in the SYSPROC schema as available to be called. The names of all procedures in the set begin with *partial-name* and can end with any string, including the empty string. For example, ABC* starts all stored procedure names that begin with ABC in the SYSPROC schema.

SCOPE
Specifies the scope of the command.

(LOCAL)
Starts the specified stored procedures in only the local members.

(GROUP)
Starts the specified stored procedures in all members of the data sharing group.

Usage notes

Errors in a definition of a stored procedure: Errors are detected at create time for a stored procedure. See CREATE PROCEDURE in Chapter 5 of *DB2 SQL Reference* for more information.

Management of stored procedures address space: The START PROCEDURE command works differently depending on how the DB2 stored procedures address spaces are managed. WLM-established stored procedure address spaces are controlled by WLM, rather than the START PROCEDURE command. For more information, see Part 5 (Volume 2) of *DB2 Administration Guide*.

Examples

Example 1: Start all stored procedures.

```
-START PROCEDURE
```

This command produces output that is similar to the following output:

```
DSNX946I - DSNX9ST2 START PROCEDURE SUCCESSFUL FOR *.*
DSN9022I - DSNX9COM '-START PROC' NORMAL COMPLETION
```

Example 2: Make the stored procedures USERPRC1 and USERPRC2 available to be called, and start any requests that are waiting for those procedures.

```
-START PROCEDURE(USERPRC1,USERPRC2)
```

This command produces output that is similar to the following output:

```
DSNX946I - DSNX9ST2 START PROCEDURE SUCCESSFUL FOR USERPRC1
DSNX946I - DSNX9ST2 START PROCEDURE SUCCESSFUL FOR USERPRC2
DSN9022I - DSNX9COM '-START PROC' NORMAL COMPLETION
```

Chapter 76. -START RLIMIT (DB2)

The DB2 command START RLIMIT starts the resource limit facility (governor) and specifies a resource limit specification table for the facility to use.

You can issue START RLIMIT even if the resource limit facility is already active. The resource limit specification table that you identify is used for new threads, and existing threads continue to be subject to the limits in the table that was active at the time they were created.

Abbreviation: -STA RLIM

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option description"
- "Example" on page 382

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

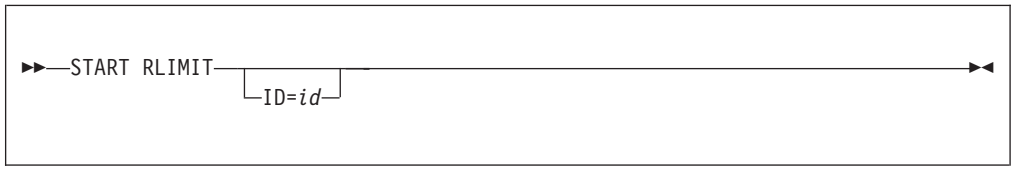
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Option description

The following keyword is optional.

© Copyright IBM Corp. 1983, 2008

ID=*id*
Identifies the resource limit specification table for the governor to use.

id is the one or two identification character that is specified when the table is created. See Part 5 (Volume 2) of *DB2 Administration Guide* for more information about resource limit specification tables.

The full name of the table is *authid*.DSNRLST*id*, where *authid* is the value that is specified in field RESOURCE AUTHID on installation panel DSNTIPP.

The **default** ID is the value that is specified in field RLST NAME SUFFIX on installation panel DSNTIPO.

Example

Start the resource limit facility.

```
-START RLIMIT ID=01
```

Chapter 77. -START TRACE (DB2)

The DB2 command START TRACE starts DB2 traces. For more information about the trace facility, see Part 5 (Volume 2) of *DB2 Administration Guide*.

An additional option for this command and additional values for a few other options exist. This additional information is intended for service and use under the direction of IBM Software Support. For details, see *DB2 Diagnosis Guide and Reference*.

Abbreviation: -STA TRA

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 384
- "Option descriptions" on page 385
- "The constraint block" on page 388
- "Usage notes" on page 393
- "Examples" on page 393

Environment

This command can be issued from a z/OS console, a DSN session, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group or local, depending on the value of the SCOPE option.

Authorization

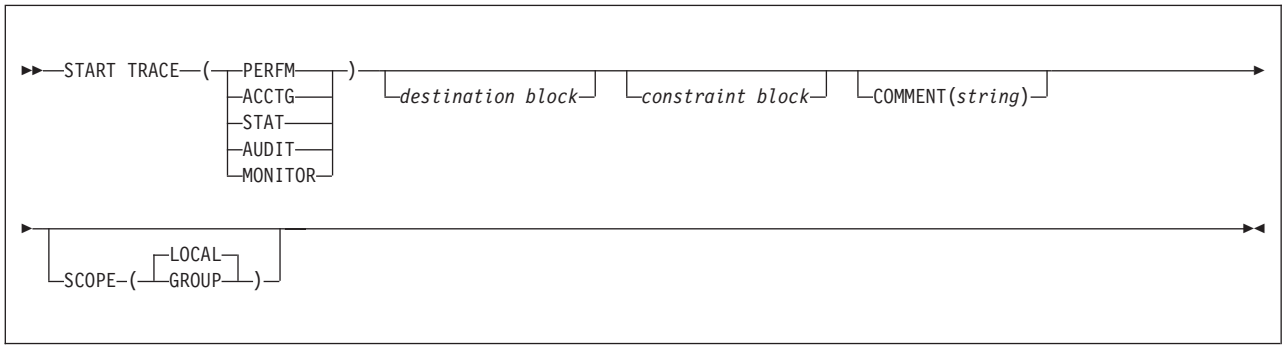
To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- TRACE privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

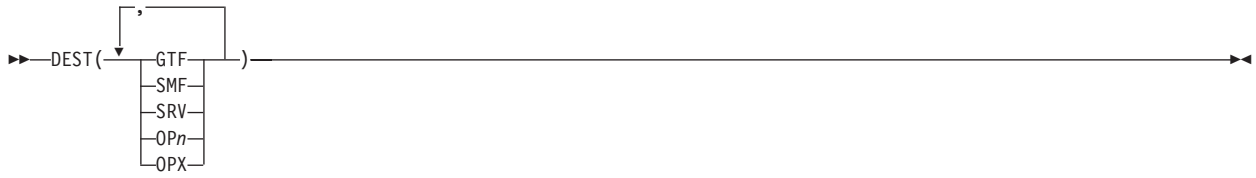
DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

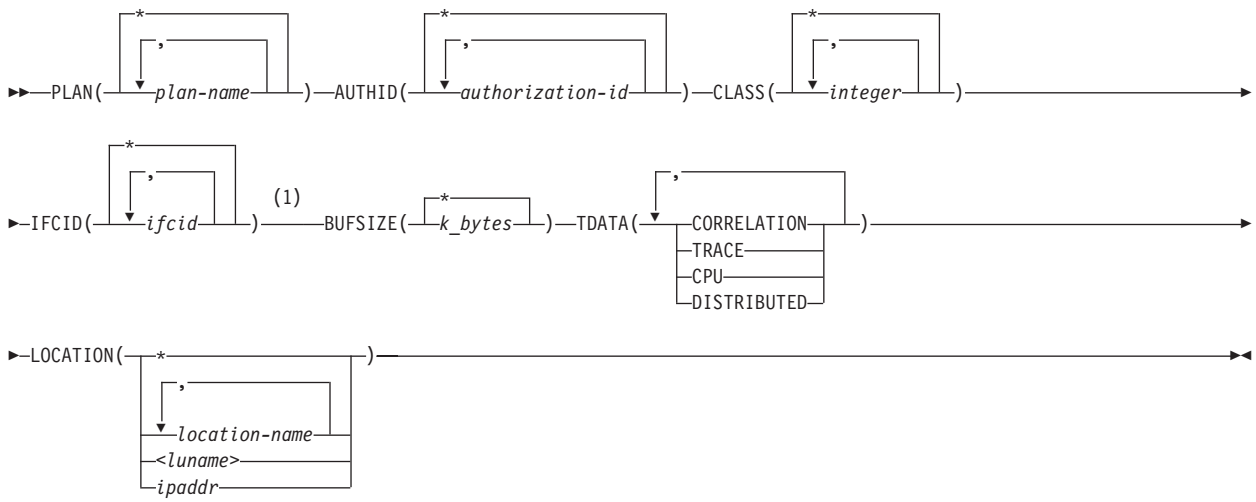
Syntax


START TRACE
(
PERFM
ACCTG
STAT
AUDIT
MONITOR
)
destination block
constraint block
COMMENT(string)
SCOPE
(
LOCAL
GROUP
)

destination block:



constraint block:



Notes:

1 Allows you to specify trace events in addition to the IFCIDs activated by the CLASS option. For details, see the description of **IFCID(** *ifcid***, ...)** on page 391.

Option descriptions

You must specify a trace type.

The options PERFM, ACCTG, STAT, AUDIT, and MONITOR identify the type of trace that is started.

(PERFM)
Specifies a trace that is intended for performance analysis and tuning. This trace includes records of specific events in the system.

Abbreviation: P

(ACCTG)
Specifies a trace that is intended to be used in accounting for a particular program or authorization ID. This trace includes records that are written for each thread.

Abbreviation: A

(STAT)
Specifies a trace that collects statistical data that is broadcast by various components of DB2, at time intervals that can be chosen during installation.

Abbreviation: S

LOCATION cannot be specified when you choose a statistics trace.

(AUDIT)
Specifies a trace that collects audit data from various components of DB2.

Abbreviation: AU

(MONITOR)
Specifies a trace that collects monitor data. This option makes trace data available to DB2 monitor application programs.

Abbreviation: MON

SCOPE
Specifies the scope of the command.

(LOCAL)
Specify to display information about procedures on the local member only.

(GROUP)
Specify to display information about procedures on all members of the data sharing group.

COMMENT(*string***)**
Gives a comment that is reproduced in the trace output (except in the resident trace tables). This option can be used to record why the command was issued.

string is any character string; it must be enclosed between apostrophes if it includes a blank, comma, or special character.

DEST
Specifies where the trace output is to be recorded. You can use more than one value, but do not use the same value twice. If you do not specify a value, the trace output is sent to the default destination shown in Table 32.

If the specified destination is not active or becomes inactive after you issue the START TRACE command, you receive message DSNW133I, which indicates that the trace data is lost. This applies for destinations GTF, SRV, and SMF. You also receive this message for destinations OP*n* and OPX if START TRACE is not issued by an application program.

Abbreviation: D

The allowable values and the default value depend on the type of trace started, as shown Table 32:

Table 32. Allowable destinations for each trace type

Type	GTF	SMF	SRV	OP*n*	OPX
PERFM	Default	Allowed	Allowed	Allowed	Allowed
ACCTG	Allowed	Default	Allowed	Allowed	Allowed

Table 32. Allowable destinations for each trace type (continued)

Type	GTF	SMF	SRV	OP*n*	OPX
STAT	Allowed	Default	Allowed	Allowed	Allowed
AUDIT	Allowed	Default	Allowed	Allowed	Allowed
MONITOR	Allowed	Allowed	Allowed	Allowed	Default

The meaning of each value is as follows:

GTF
The z/OS generalized trace facility (GTF). The record identifier for records from DB2 is X'0FB9'.

SMF
The system management facility. The SMF record type of DB2 trace records depends on the IFCID record, as follows:

IFCID record	SMF record type
1 (System Services Statistics)	100
2 (Database Services Statistics)	100
3 (Agent Accounting)	101
202 (Dynamic System Parameters)	100
230 (Data Sharing Global Statistics)	100
239 (AGENT ACCOUNTING OVERFLOW)	101
All Others	102

SRV
An exit to a user-written routine. For instructions and an example of how to write such a routine, see the macro DSNWVSER in library *prefix*.SDSNMACS.

OP*n*
A specific destination.

n can be an integer from 1 to 8.

OPX
A generic destination which uses the first free OP*n* slot.

Only applications that start a trace to an OP*n* buffer can read that buffer. For more information on starting a trace via an application program, see Appendix E (Volume 2) of *DB2 Administration Guide*.

All traces to an OPX destination must be stopped before the buffer is marked as not in use. Traces that are started to an OPX buffer that was formerly in use write over the storage any previous traces had set.

The constraint block

The constraint block places optional constraints on the kinds of data that are collected by the trace. The allowable constraints depend on the type of trace started, as shown Table 33:

Table 33. Allowable constraints for each trace type

Type	PLAN	AUTHID	CLASS	LOCATION
PERFM	Allowed	Allowed	Allowed	Allowed
ACCTG	Allowed	Allowed	Allowed	Allowed
STAT	NO	NO	Allowed	NO
AUDIT	Allowed	Allowed	Allowed	Allowed
MONITOR	Allowed	Allowed	Allowed	Allowed

The meaning of each option is as follows:

PLAN(*plan-name*, **...)**
Introduces a list of specific plans for which trace information is gathered. You cannot use this option for a STAT trace.

The **default** is **PLAN(*).**

(*) Starts a trace for all plans.

plan-name
Is the name of an application plan. You can use up to eight names; a separate trace is started for each name. If you use more than one name, you can use only one value for AUTHID and LOCATION.

AUTHID(*authorization-id*, **...)**
Introduces a list of specific authorization IDs for which trace information is gathered. The authorization IDs specified must be the primary authorization IDs. You cannot use this option for a STAT trace.

The **default** is **AUTHID(*).**

(*) Starts a trace for all authorization IDs.

authorization-id
Specifies an authorization ID. You can use up to eight identifiers; a separate trace is started for each identifier. If you use more than one identifier, you can use only one value for PLAN and LOCATION.

CLASS(*integer*, **...)**
Introduces a list of classes of data gathered. What classes are allowable, and their meaning, depends on the type of trace started.

Abbreviation: C

When this option is omitted, all the default classes within the trace type are activated. The **default** classes for each trace type are marked by asterisks (*) in Table 34 on page 389.

(*) Starts a trace for all classes of the trace type.

integer
Is any number in the following table. You can use any number of classes that are allowed for the type of trace started.

Table 34. Classes for DB2 trace types

Class	Description of class	Activated IFCIDs
Accounting trace		
1*	Standard accounting data	3,106,200,239
2	Entry or exit from DB2 event signalling	232
3	Elapsed wait time in DB2	6-9,32,33,44,45,117,118,127,128,170,171, 174,175,213-216,226,227,242,243,321,322,329
4	Installation-defined accounting record[1]	151
5	Time spent processing IFI requests	187
6	Reserved	
7	Entry or exit from DB2 event signalling for package and DBRM accounting	232,240
8	Wait time for a package	6-9,32,33,44,45,117,118,127,128,170,171, 174,175,213-216,226,227,241-243,321,322
10	Package detail	239
11 - 29	Reserved	
30 - 32	Available for local use	
Audit trace		
1*	Access attempts denied due to inadequate authorization	140
2	Explicit GRANT and REVOKE	141
3	CREATE, ALTER, and DROP operations against audited tables	142
4	First change of audited object	143
5	First read of audited object	144
6	Bind time information about SQL statements that involve audited objects	145
7	Assignment or change of authorization ID	55,83,87,169,319
8	Utilities	23,24,25,219,220
9	Installation-defined audit record[1]	146
10 - 29	Reserved	
30 - 32	Available for local use	
Statistics trace		
1*	Statistics data	1,2,105,106,202
2	Installation-defined statistics record[1]	152
3	Deadlock, lock escalation, group buffer pool, data set extension information, and indications of long-running URs and active log space shortages	172,196,250,258,261,262,313,330,337
4	DB2 exceptional conditions	173,191-195,203-210,235,236,238,267,268
5	DB2 data sharing statistics record	230
6	Storage usage details	225
7	Reserved	
8	Data set I/O statistics	199
9 - 29	Reserved	

Table 34. Classes for DB2 trace types (continued)

Class	Description of class	Activated IFCIDs
30 - 32	Available for local use	
Performance trace		
1*	Background events	1,2,31,42,43,76-79,102,103,105-107,153
2*	Subsystem events	3,68-75,80-89,106,174,175
3*	SQL events	22,53,55,58-66,92,95-97,106,112,173,177, 233,237,250,272,273,325
4	Reads to and writes from the buffer and EDM pools	6-10,29-30,105-107,127,128,226,227,321,322
5	Write to log; archive log	32-41,104,106,114-120,228,229
6	Summary lock information	20,44,45,105-107,172,196,213,214,218,337
7	Detailed lock information	21,105-107,223
8	Data scanning detail	13-18,105-107,125,221,222,231,305,311
9	Sort detail	26-28,95-96,106
10	BIND, commands, and utilities detail	23-25,90,91,105-107,108-111,201,256
11	Execution unit switch and latch contentions	46-52,56,57,93,94,106,113
12	Storage manager	98-101,106
13	Edit and validation exits	11,12,19,105-107
14	Entry from and exit to an application	67,106,121,122
15	Installation-defined performance record[1]	154
16	Distributed processing	157-163,167,183
17	Claim and drain information	211-216
18	Event-based console messages	197
19	Reserved	
20	Data sharing coherency summary	249-251,256-257,261,262,267,268
21	Data sharing coherency detail	255,259,263
22	Authorization exit parameters	314
23 - 29	Reserved	
30 - 32	Available for local use	
Monitor trace		
1*	Standard accounting data	200
2	Entry or exit from DB2 event signalling	232
3	DB2 wait time for I/O, locks; resource usage information	6-9,32,33,44,45,117,118,127, 128,170,171,174,175,213,214, 215,216,226,227,242,243,321,322
4	Installation-defined monitor record[1]	155
5	Time spent processing IFI requests	187
6	Changes to tables created with DATA CAPTURE CHANGES	185
7	Entry or exit from DB2 event signalling for package and DBRM accounting	232,240

Table 34. Classes for DB2 trace types (continued)

Class	Description of class	Activated IFCIDs
8	Wait time for a package	6-9,32,33,44,45,51,52,56,57, 117,118,127,128,170,171,174, 175,213-216,226,227,241-243,321,322
10	Package detail	239
9, 11- 29	Reserved	
30 - 32	Available for local use	

Note:

- An asterisk (*) indicates a default class for a trace type.
- 1. For instructions on using the IFCIDs, see Appendix D (Volume 2) of *DB2 Administration Guide*.
- 2. DB2 does not collect statistical data for this record unless IFCID 318 is activated. IFCID 318 is not associated with any trace class; you must start it on its own.

IFCID(*ifcid*, ...)
Specifies which other IFCIDs (trace events), in addition to those IFCIDs contained in the classes specified in the CLASS option, are to be started. To start only those IFCIDs specified in the IFCID option, use trace classes 30-32. These classes have no predefined IFCIDs and are available for a location to use. (See "Example 1" on page 394 for an example of activating only those trace events specified in the IFCID option.)

If you do not specify the IFCID option, only those IFCIDs contained in the activated trace classes are started.

The maximum number of IFCIDs is 156. The range of values that are valid for the IFCID option is 1 through 350, with the exception of: 4, 5, 185, 187, 217, 232, 234, 240, and 241. These exceptions are invalid values for the IFCID option. IFCIDs 4 and 5 are always automatically active. Some of the other invalid IFCIDs can be activated only by certain trace classes. The invalid values for the IFCID option that can be started only by trace classes are:

To start...	Start...
IFCID 185	monitor trace class 6
IFCID 232	monitor trace class 2 or 7, or accounting trace class 2 or 7
IFCID 240	monitor trace class 7 or accounting trace 7
IFCID 241	monitor trace class 8 or accounting trace 8

The **default** is **IFCID(*)**.

BUFSIZE(*k_bytes*, ...)
Specifies the size of an IFC managed buffer that receives the trace data. You can specify this option only if you specified an OP*n* destination.

k_bytes can range from 256 KB to 16384 KB. The number must be evenly divisible by 4. If you specify a value outside of this range, the range limit closest to the specified value is used. To allocate a buffer size of 256 KB, you would specify BUFSIZE(256).

The **default** is **BUFSIZE(*)**, which is the size set when DB2 was installed.

TDATA
Specifies the product section headers to be placed into the product section of each trace record. If you do not specify TDATA, then the type of trace determines the type of product section header. The product section of a trace record can contain multiple headers.

All IFC records have a standard IFC header. The correlation header is added for accounting, performance, audit, and monitor records. The trace header is added for serviceability records.

CORRELATION
Places a correlation header on the record.

Abbreviation: COR

TRACE
Places a trace header on the record.

Abbreviation: TRA

CPU
Places a CPU header on the record. The CPU header contains the current processor time for the z/OS TCB or SRB executing.

DISTRIBUTED
Places a distributed header on the record.

Abbreviation: DIST

LOCATION(*location-name***, ...)**
Introduces a list of specific location names for which trace information is gathered. The use of the LOCATION option precludes tracing threads that have no distributed data relationship. LOCATION cannot be specified when you want to start a statistics trace.

location-name
Identifies the DB2 subsystems whose distributed threads you want to trace. Activates the DB2 trace for the remote TCP/IP or SNA location that you specify by *location-name*.

You can specify up to eight locations; a separate trace is started for each one. You can specify only one location if you use more than one plan name or authorization ID.

<luname>
Activates the DB2 trace for the remote clients that are connected to DDF through the remote SNA LU name that you specified in *luname*.

ipaddr
Activates the DB2 trace for the remote clients that are connected to DDF through the remote TCP/IP host.*nnn.nnn.nnn*is the dotted decimal IP address.

(*) Indicates that you want to start trace events that occur under distributed threads regardless of which location they are connected to. Specifying the local location name is equivalent to specifying LOCATION(*).

Clients other than DB2 UDB for z/OS: DB2 UDB for z/OS does not receive a location name from clients that are not DB2 UDB for z/OS subsystems. To start a trace for a client that is not a DB2 UDB for z/OS subsystem, enter its LUNAME or IP address. Enclose the LUNAME by the less-than (<) and greater-than (>) symbols. Enter the IP address in the form *nnn.nnn.nnn.nnn*. For example, to start a trace for a client with the LUNAME of LULA, enter the following command:

```
-START TRACE (PERFM) CLASS (*) LOCATION (<LULA>)
```

To start a trace for a client with the IP address of 123.34.101.98, enter the following command:

```
-START TRACE (PERFM) CLASS (*) LOCATION (123.34.101.98)
```

Usage notes

Number of traces: If you use one or no values for PLAN, AUTHID, or LOCATION, the START TRACE command starts a single trace. If you use multiple values for PLAN, AUTHID, or LOCATION, the command starts a trace for each plan, authorization ID, or location. There can be up to 32 traces going at one time. If a START TRACE command is entered from the console or from the DB2I panels to an OP*n* or an OPX destination, message DSNW133I is issued to indicate trace data lost.

Using the options PLAN, AUTHID, or LOCATION when starting monitor trace class 1 has no effect on the amount of data returned on IFI READS requests. See Appendix E (Volume 2) of *DB2 Administration Guide* for more information on qualifying monitor trace class 1 IFCIDs.

Using the options PLAN, AUTHID, or LOCATION has no effect when starting either accounting or monitor trace classes 2, 5, or 7.

Stopping and starting DB2: If DB2 is stopped and started after you have started a trace, the trace is not restarted automatically.

| **Specifying SCOPE (GROUP):** When you issue START TRACE with
| SCOPE(GROUP), DB2 issues a START TRACE command on each member of the
| data sharing group. The data goes to the destination as it is defined for each
| member of the data sharing group. If you want to gather trace data for all
| members of the data sharing group in one place, use a monitor program with IFI
| READA or READS calls to collect the data. See Appendix E (Volume 2) of *DB2*
| *Administration Guide* for more information.

| If a trace is started with SCOPE(GROUP), and a new member joins the data
| sharing group after the trace is started, the new member also writes the trace data
| that is specified by the START TRACE command.

| Starting a trace with SCOPE(GROUP) can generate large amounts of trace data, so
| you might need to increase the size of the return area in your monitor program to
| hold the extra data.

Trace destination precedence: If an IFCID is associated with a class, and you
specify that IFCID in the IFCID keyword, the destination for the class takes
precedence. This rule affects IFCIDs in accounting classes 2, 3, 5, 7, 8, and
monitoring classes 1, 2, 3, 5, 7, 8 because these classes have preset destinations for
the IFCIDs.

For example, the following command ″-START TRACE(ACCTG) CLASS (1,2,3)
IFCID(6,7) DEST SMF,″ will not write IFCIDs 6 and 7 to SMF because they are part
of accounting class 3, which has a preset destination. To write IFCIDs 6 and 7 to
SMF, you need to start the trace as follows:

```
-START TRACE(ACCTG) CLASS(1,2,3) DEST(SMF)
-START TRACE(ACCTG) CLASS(30) IFCID(6,7) DEST(SMF)
```

Examples

Example 1: Start a performance trace for threads with remote activity to location USIBMSTODB21. Only activate IFCIDs 44 (lock suspends) and 54 (lock contention). Trace class 30 is available for installation use.

```
-START TRACE (PERFM)
  DEST(GTF)
  LOCATION(USIBMSTODB21)
  CLASS(30)
  IFCID(44)
```

Example 2: Start an accounting trace for plan DSN8BC81. Write records to SMF (that will happen by default). Include a comment to identify the trace.

```
-START TRACE (ACCTG)
  PLAN (DSN8BC81)
  COMMENT ('ACCTG TRACE FOR DSN8BC81')
```

Example 3: Start the statistics trace. Write records to SMF (by default).

```
-START TRACE=S
```

Example 4: Start monitor tracing (usually done by an application program). Write records to OPX (by default).

```
-START TRACE(MON)
```

| ***Example 5:*** Start monitor tracing (usually done by an application program) on the
| data sharing group. Write records to OPX (by default).

```
-START TRACE(MON) SCOPE(GROUP)
```

Chapter 78. /STOP (IMS)

The IMS /STOP command (with the SUBSYS parameter) prevents application programs from accessing external subsystem resources.

The following is only a partial description of the /STOP command. For a complete description, see *IMS Command Reference.*

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage note" on page 396

Environment

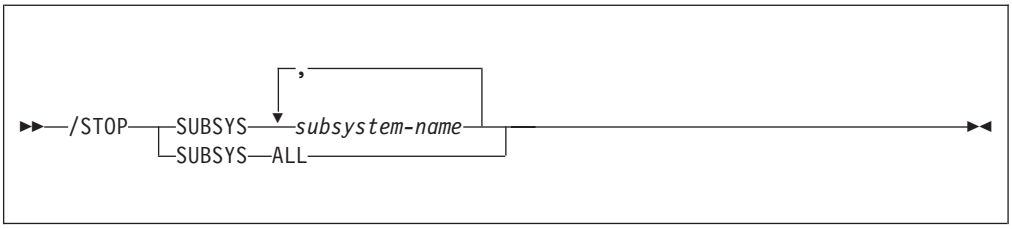
This command can be issued only from an IMS terminal.

Data sharing scope: Member

Authorization

This command requires an appropriate level of IMS authority, as described in the *IMS Administration Guide: System.*

Syntax



Option descriptions

SUBSYS
Specifies whether connection is to be stopped for one or more names of external subsystems presently connected to IMS, or for all of them.

subsystem-name, **...**
Specifies one or more names of external subsystems whose connection to IMS is to be stopped.

ALL
Indicates that connection is to be stopped for all external subsystems presently connected to IMS.

© Copyright IBM Corp. 1983, 2008

Usage note

When to use /STOP: The /STOP command allows application programs currently accessing external resources to complete normally. When all applications have terminated, the connection to the external subsystem is also terminated. A /START command must be issued to reestablish the connection.

The /STOP command can also be used to stop the subsystem connection in order to change the specifications in the external subsystem's PROCLIB member entry. The /START command then refreshes the copy in main storage of the PROCLIB entry with the modified entry.

Chapter 79. -STOP DATABASE (DB2)

The DB2 command STOP DATABASE makes the specified objects unavailable for applications and closes their data sets. The objects that can be designated are:

- Databases
- Table spaces
- Index spaces
- Physical partitions of partitioned table spaces or index spaces (including index spaces that contains DPSIs)
- Logical partitions of nonpartitioned secondary indexes

When used to stop a logical partition of a secondary index, the command does not close any data sets that are associated with the index.

In a data sharing environment, the command applies to every member of the data sharing group. If a GBP-dependent object is stopped with the command STOP DATABASE, DB2 performs the necessary processing to make the object no longer GBP-dependent.

Abbreviation: -STO DB

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 398
- "Option descriptions" on page 398
- "Usage notes" on page 400
- "Examples" on page 402

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group

Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- STOPDB privilege
- DBMAINT authority
- DBCTRL authority
- DBADM authority
- SYSCTRL authority
- SYSADM authority

Error messages are produced for those specified databases for which this set does not have the STOPDB privilege.

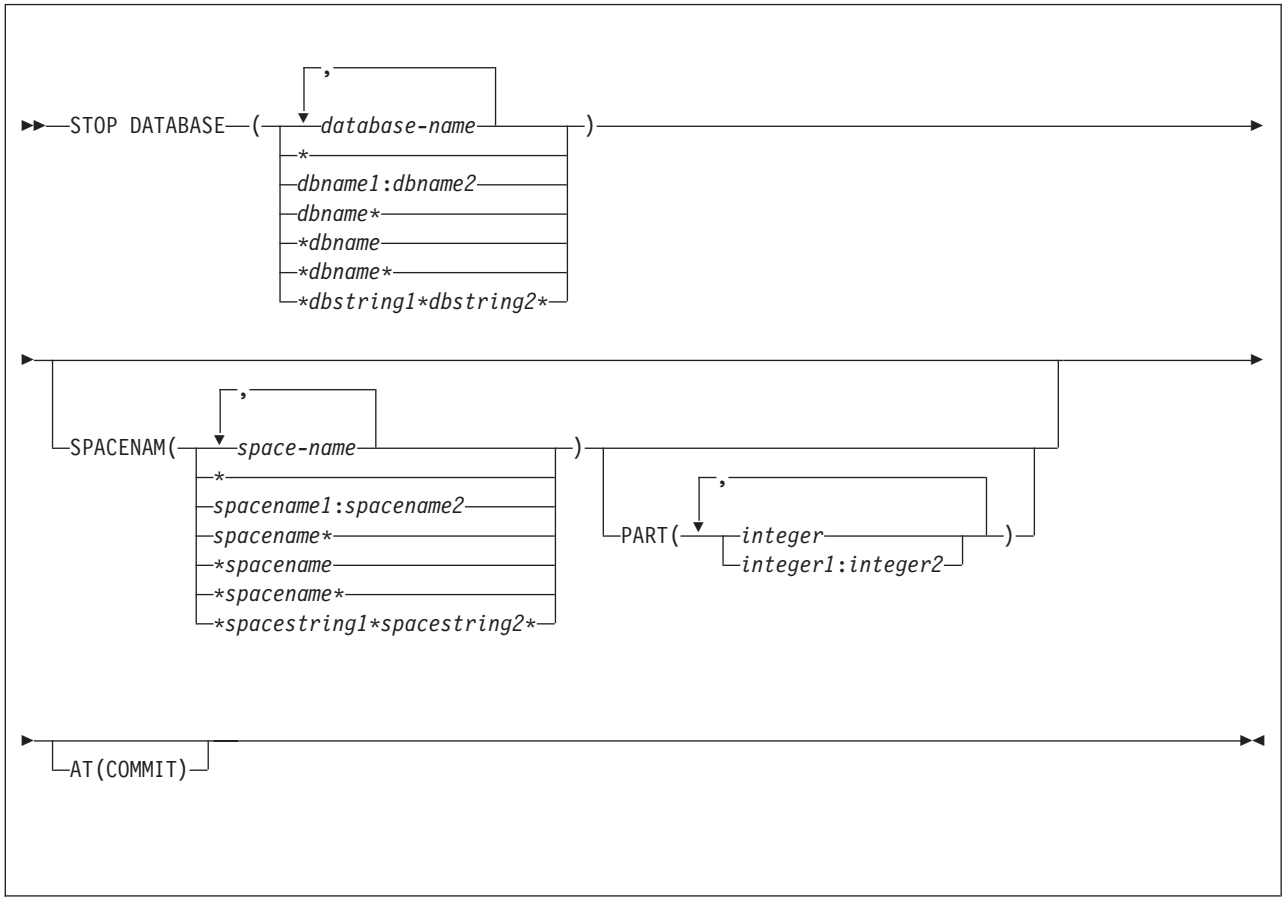
© Copyright IBM Corp. 1983, 2008

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

When data definition control is active, installation SYSOPR or installation SYSADM authority is required to stop the database, a table space, or an index space that contains a registration table or index.

Database DSNDB06 contains the table spaces and index spaces that are required to check authorization. If you stop any table space or index space that is required for the START DATABASE authorization check, installation SYSADM authority is required to restart it.

Syntax



Option descriptions

One of the following two options is required.

(*database-name***, ...)**
Specifies the names of the database, or database for the table spaces or index spaces to stop. If you use more than one name, separate names in the list by commas.

(*) Stops all databases for which the privilege set of the process has at least DBMAINT authority or STOPDB privilege.

However, DSNDB01, DSNDB06, DSNRTSDB, and work file databases, such
as DSNDB07, can be stopped only by specifying them explicitly (for
example, STOP DATABASE(DSNDB01)).

dbname and *dbstring* can have any of the forms in the following list (where *dbname1* and *dbname2* represent any strings of from 1 to 8 characters, and *dbname* represents any string of from 1 to 7 characters):

Form	Stops...
dbname1:dbname2	All databases whose names collate greater than or equal to *dbname1* and less than or equal to *dbname2*
*dbname**	All databases whose names begin with the string *dbname*
**dbname*	All databases whose names end with the string *dbname*
dbname	All databases whose names contain the string *dbname*
dbstring1*dbstring2	All databases whose names contain the strings *dbstring1* and *dbstring2*

SPACENAM(*space-name***, ...)**
Indicates names of table spaces or indexes within the specified database to stop.

Abbreviation: SPACE, SP

space-name
Is the name of one or more table spaces or index spaces to stop.

You can write *space-name* like *database-name* to designate:
- The name of a single table space or index space
- A range of names
- A partial name, including a beginning or ending pattern-matching character (*), pattern-matching character between two strings, or any combination of these uses. Consecutive pattern-matching characters (*) are not allowed, and you cannot specify two pattern-matching characters in the middle of a keyword string.

See "Usage notes" on page 400 for instructions on how to start a table space or index space again.

(*) Stops all table spaces and indexes of the specified database.

spacename and *spacestring* can have any of the forms in the following list (where *spacename1* and *spacename2* represent any strings of from 1 to 8 characters, and *spacename* represents any string of from 1 to 7 characters):

Form	Displays the status of...
spacename1:spacename2	All table spaces or index spaces whose names collate greater than or equal to *spacename1* and less than or equal to *spacename2*
*spacename**	All table spaces or index spaces whose names begin with the string *spacename*

**spacename*	All table spaces or index spaces whose names end with the string *spacename*
spacename	All table spaces or index spaces whose names contain the string *spacename*
spacestring1*spacestring2	All table spaces or index spaces whose names contain the strings *spacestring1* and *spacestring2*

PART (*integer*, ...)
Indicates the partition number of one or more partitions, within the specified table space or index, that are to be stopped. The START or STOP state of other partitions does not change.

The *integer* specified must identify a valid partition number for the corresponding space name and database name. If you specify nonvalid partition numbers, you receive an error message for each nonvalid number, but all valid partitions that you specified are stopped.

integer can be written to designate one of the following specifications:

- A list of one or more partitions
- A range of all partition numbers that collate greater than or equal to *integer1* and less than or equal to *integer2*
- A combination of lists and ranges

PART is valid with partitioned table spaces, partitioned indexes, and nonpartitioned type 2 indexes of partitioned table spaces. If you specify PART with a nonpartitioned table space or index on a nonpartitioned table space, you receive an error message, and the nonpartitioned space is not stopped. When a logical partition is stopped, the index is not closed. A nonpartitioning index must be stopped without the use of PART to close the index.

AT(COMMIT)
Marks the specified object as being in STOP status to prevent access from new requesters. Currently running applications are allowed to continue access until their next commit. After commit, further access by the committing application is prohibited. The object is actually stopped and put in STOP status when all jobs release their claims on it and all utilities release their drain locks on it. Specify AT(COMMIT) to break in on threads that are bound with RELEASE(DEALLOCATE), especially in situations where there is high thread reuse.

The option is ignored for declared temporary databases and table spaces within it.

Usage notes

Explicitly stopped databases: If table spaces and indexes are stopped explicitly (using the STOP DATABASE command with the SPACENAM option), they must be started explicitly using the START DATABASE command. Starting the database does not start table spaces or indexes that have been stopped explicitly.

Stopped table spaces, indexes, and partitions: Table spaces, indexes, and partitions are physically closed when the STOP DATABASE command is issued, except for logical partitions of a nonpartitioning index of a partitioned table space. Index spaces for declared temporary tables cannot be stopped or started.

Operation in TSO, z/OS, and batch: When the STOP DATABASE command is issued from a TSO or a z/OS console, the command operates asynchronously to keep the terminal free. When the command is issued from a batch job, it operates synchronously in case later steps depend on the database being stopped. The STOP DATABASE command drains work in progress on the database before stopping it. If it cannot get the drain locks on the first request, it repeatedly tries again. The command fails if it times out more than 15 times trying to get the locks or if a serious deadlock situation occurs.

Ensuring that all databases are stopped: When the STOP DATABASE command is processing asynchronously, message DSN9022I might be issued before the command completes. Message DSNT736I is issued to indicate that the asynchronous processing of the STOP DATABASE command is complete.

Use the DISPLAY DATABASE command to check the stopped status of table spaces and indexes in a database. A status of STOPP indicates that the object is in the process of being stopped. A status of STOP indicates that the stop has completed and the object is in a stopped state. An object is not stopped until all currently active threads accessing the object are quiesced.

An object might remain in the STOP pending (STOPP) status if the STOP DATABASE command does not successfully complete processing.

Stopping the communication database and the resource limit database: If the communication database (CDB) and the resource limit database (RLST) are active, they cannot be stopped. Those databases are active when created and are activated by DB2. For more information on the CDB, see Part 3 of *DB2 Installation Guide*. For more information about the RLST, see Part 5 (Volume 2) of *DB2 Administration Guide*.

Stopping DSNDB01: If you try to stop the DSNDB01 database while an application plan or package is executing, you might receive a time out because of locking contention on DSNDB01. This is most likely to occur when an application plan or package is executing for the first time since DB2 was started, or if the skeleton cursor table (SKCT) for the plan or the skeleton package table (SKPT) for the package was swapped out of the EDM pool.

Table space in a restrictive status: If an application process requests a transaction lock on a table space that is in a restrictive status (RECP) or has a required index in a restrictive status, DB2 acquires the lock and does not detect the status until the application tries to access the table space or index. The application then receives SQLCODE -904 (“resource not available”) and should release the lock, either by committing or rolling back (if the value of the RELEASE option is COMMIT) or by ending (if the value of RELEASE is DEALLOCATE). If you issue the command STOP DATABASE for either the table space or the index space while a transaction lock is in effect, the command is suspended. It repeatedly tries to get the locks needed to drain the work in progress before stopping the database. If the command times out more than 15 times trying to get the locks, it fails.

After a disk failure: Issuing the STOP DATABASE command before interrupting the I/O interface between the failed device and DB2 can result in incomplete I/O requests. To prevent this hang situation, create an interruption either by forcing the device offline using the z/OS command VARY with the FORCE option, or by setting the I/O timing interval for the device before any failures. You can set the I/O timing interval through the IECIOSxx z/OS parmlib member or by issuing the z/OS command:

```
SETIOS MIH,DEV=dddd,IOTIMING=mm:ss
```

Stopping a LOB table space: The STOP DATABASE command can be used to stop LOB table spaces and indexes on auxiliary tables. LOB table spaces are stopped independently of the base table space with which the LOB table space is associated.

The following table summarizes the locking used by the STOP DATABASE command.

Table 35. Locking used by the STOP DATABASE command

Command	Table space type		Locks acquired
STOP AT COMMIT	*Partitioned*	PART	IX mass delete lock. Drain-all on partitions specified.
			IX mass delete lock. Drain-all on all partitions.
	Nonpartitioned		IX mass delete lock. Drain-all on table space.
STOP	*Partitioned*	PART	X-lock partitions specified. Drain-all on partitions specified.
			X-lock all partitions. Drain-all on all partitions.
	Nonpartitioned		X-lock table space. Drain-all on table space.

Examples

Example 1: Stop table space DSN8S81E in database DSN8D81A and close the data sets that belong to that table space.

```
-STOP DATABASE(DSN8D81A) SPACENAM(DSN8S81E)
```

Example 2: Stop all databases (except DSNDB01, DSNDB06, and work file databases)

```
-STOP DATABASE(*)
```

Example 3: Stop all databases (except DSNDB01, DSNDB06, and work file databases) when all jobs release their claims and all utilities release their drain locks.

```
-STOP DATABASE(*) AT(COMMIT)
```

Example 4: Stop the first partition of XEMP2, a nonpartitioning index of a partitioned table space in database DSN8D81A. Partition 1 is logically stopped and cannot be accessed by applications; however, no data sets are closed because parts of a nonpartitioning index are not associated with separate physical data sets.

```
-STOP DATABASE(DSN8D81A) SPACENAM(XEMP2) PART(1)
```

Example 5: Stop all table spaces with names that begin with "T" and end with the "IQUA03" string in database DSN8D81A.

```
-STOP DATABASE(DSN8D81A) SPACENAM(T*IQUA03)
```

Output similar to the following output indicates that the command completed successfully:

```
DSN9022I - DSNTDDIS 'STOP DATABASE' NORMAL COMPLETION
DSNT736I - ASYNCHRONOUS STOP DATABASE COMMAND HAS
COMPLETED FOR COMMAND: STOP DB(DSN8D81A) SPACE(T*IQUA03)
```

Chapter 80. -STOP DB2 (DB2)

The DB2 command STOP DB2 stops the DB2 subsystem.

Abbreviation: -STO DB2

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Usage notes" on page 404
- "Example" on page 404

Environment

This command can be issued from a z/OS console, a DSN session, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

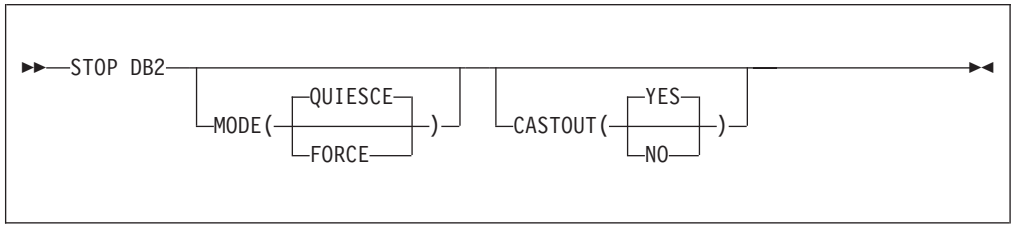
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- STOPALL privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Option descriptions

MODE

Indicates whether currently executing programs will be allowed to complete. For the effects of this option on distributed threads, see the description of the MODE option of Chapter 81, "-STOP DDF (DB2)," on page 405.

© Copyright IBM Corp. 1983, 2008

(QUIESCE)
Allows currently executing programs to complete processing. No new program is allowed to start.

(FORCE)
Terminates currently executing programs, including utilities. No new program is allowed to start. MODE(FORCE) probably causes indoubt situations. Some tasks, such as stored procedures tasks and DB2 service tasks, terminate abnormally. When they terminate abnormally, you might see dumps and messages from these failures.

CASTOUT
Specifies whether the DB2 member performs castout processing for the page sets or partitions for which the member was last updated. The CASTOUT option only applies in a data sharing environment.

YES
Allow group buffer pool castout processing.

NO
Skip group buffer pool castout processing.

Usage notes

MODE(QUIESCE): If MODE(QUIESCE) is used, all connected address spaces must terminate all connections before the DB2 subsystem stops. The system operator can tell whether any connections remain by using the DISPLAY THREAD command, and can cancel them by using the DB2 CANCEL command or z/OS commands.

MODE(FORCE): A forced stop does not cause an immediate abend. If a connected task is executing outside DB2, DB2 posts an exit routine to stop the task from accessing DB2. If a task is executing in DB2, it stops when the next "suspend" or "execution unit switch" occurs. In some cases, the delay before stopping can be significant.

CASTOUT(NO): Consider using CASTOUT(NO) when shutting down a DB2 data sharing member for maintenance, because the option can speed shutdown processing in a data sharing environment. If you are shutting down multiple members of a data sharing group with CASTOUT(NO), some changed data might reside in the group buffer pools after the members have shut down. Therefore, if you want consistent data on disk (for example, you are shutting down all members to create a copy of the database to send offsite), do not use CASTOUT(NO).

With CASTOUT(NO), the DB2 member shuts down with QC status, as displayed by the DISPLAY GROUP command, which indicates that the member quiesced with some castout processing not completed. A retained page set or partition P-lock is held in IX state for each object for which the DB2 member was the last updater. Also, group buffer pool connections enter a failed-persistent state.

Example

Example 1: Stop the DB2 subsystem. Allow currently active programs to complete. Do not allow new programs to identify to DB2.

```
-STOP DB2 MODE (QUIESCE)
```

Example 2: Stop a member of a data sharing group for maintenance.

```
-STOP DB2 MODE (QUIESCE) CASTOUT(NO)
```

Chapter 81. -STOP DDF (DB2)

The DB2 command STOP DDF stops the distributed data facility (DDF) if it has already been started; use this command to terminate the DDF interface to VTAM or TCP/IP.

Abbreviation: -STO DDF

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions" on page 406
- "Usage notes" on page 406
- "Examples" on page 407

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

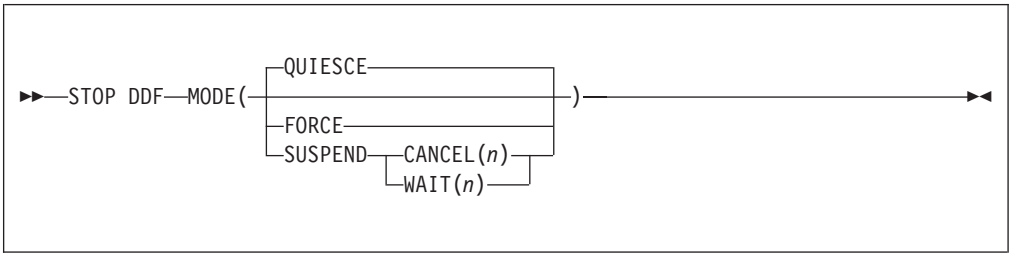
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



© Copyright IBM Corp. 1983, 2008

Option descriptions

MODE
Indicates whether currently executing active distributed threads are allowed to complete.

(QUIESCE)
Allows active distributed threads that are using DDF to complete normally and terminates only inactive distributed threads. If DDF THREADS ACTIVE was specified during DB2 installation, all DDF threads are active threads.

(FORCE)
Terminates all currently executing distributed threads.

Some tasks, such as stored procedures tasks and DB2 service tasks, terminate abnormally. When they terminate abnormally, you might see dumps and messages resulting from these failures.

(SUSPEND)
Suspends all DDF threads by:

- Keeping inactive DDF threads inactive until a subsequent START DDF command is issued
- Terminating all DDF pool threads
- Preventing inbound DDF work from starting

MODE(SUSPEND) is intended to be used at a DB2 DRDA server when locking conflicts exist between CREATE, ALTER, DROP, GRANT, or REVOKE operations and client access to data. Requests that normally cause work to be dispatched (including requests for new connections) are queued. Outbound DDF processing is not affected by this command.

CANCEL (*n*)
Cancels all active DDF database access threads if suspend processing does not complete in *n* seconds. The range of *n* is 0 to 9999.

WAIT (*n*)
Resumes DDF processing if suspend processing does not complete in *n* seconds. The range of *n* is 0 to 9999.

Usage notes

MODE(QUIESCE): If MODE(QUIESCE) is used, all distributed activity must complete before DDF stops. The operator can tell whether any distributed threads remain by using DISPLAY THREAD with the LOCATION option. To cancel distributed threads that are preventing DDF from stopping, see “Usage notes” on page 98 for CANCEL THREAD, or use STOP DDF MODE(FORCE).

MODE(QUIESCE) forces any inactive threads to terminate. A requesting system that is using two-phase commit on an inactive thread might report the terminated thread as indoubt at the system that issued STOP DDF. The thread is not actually indoubt (no commit or rollback is pending), and the condition is resolved when DDF is restarted.

MODE(FORCE): If MODE(FORCE) is used, the DB2 connection to VTAM or TCP/IP terminates. The termination forces all VTAM or TCP/IP requests to complete immediately, indicating that a communications error has occurred and DDF has stopped. A forced stop might take as long as three minutes to complete.

If any applications are updating remote servers that use two-phase commit, MODE(FORCE) might result in indoubt threads at each server.

MODE(SUSPEND): If MODE(SUSPEND) completes successfully, additional database resources, which are not inbound DDF work, might still be held. Cancel these additional resources with CANCEL THREAD as described in Chapter 16, "-CANCEL THREAD (DB2)," on page 97.

Table 36 summarizes the actions that DB2 takes when START DDF, STOP DDF, START DB2, and STOP DB2 commands are issued with different DDF states.

Table 36. The result of commands on the DDF status

DDF status	START DDF command	STOP DB2 or STOP DDF command without MODE(FORCE)	STOP DB2 or STOP DDF command with MODE(FORCE)	STOP DDF command with MODE(SUSPEND)
Starting	DSNL003I	DSNL003I	DSNL003I	DSNL003I
Started	DSNL001I	DDF stops	DDF forced stop	DDF suspends
Stopping	DSNL005I	DSNL005I	DSNL005I	DSNL005I
Stopped	DDF starts	DSNL002I	DSNL002I	DSNL002I
Suspending	DDF resumes	DDF stops	DDF forced stop	DSNL069I
Suspended	DDF resumes	DDF stops	DDF forced stop	DSNL065I

Examples

Example 1: Stop the distributed data facility (MODE QUIESCE).

```
-STOP DDF
```

Example 2: Stop the distributed data facility (MODE FORCE).

```
-STOP DDF MODE(FORCE)
```

Example 3: Suspend distributed data facility activity (MODE SUSPEND). If command processing continues after 600 seconds, cancel any remaining DDF threads.

```
-STOP DDF MODE(SUSPEND) CANCEL(600)
```

Chapter 82. -STOP FUNCTION SPECIFIC (DB2)

The DB2 command STOP FUNCTION SPECIFIC prevents DB2 from accepting SQL statements with invocations of the specified functions. This command does not prevent SQL statements with invocations of the functions from running if they have already been queued or scheduled by DB2. You cannot use this command to stop built-in functions or user-defined functions that are sourced on another function.

DB2 implicitly issues the command STOP FUNCTION SPECIFIC ACTION(REJECT) for any function that exceeds the maximum abend count. That count is set by the MAX ABEND COUNT field of installation panel DSNTIPX.

Abbreviation: -STO FUNC SPEC

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 410
- "Option descriptions" on page 410
- "Usage notes" on page 411
- "Examples" on page 411

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group or local, depending on the value of the SCOPE option.

Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities for each function:

- Ownership of the function
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

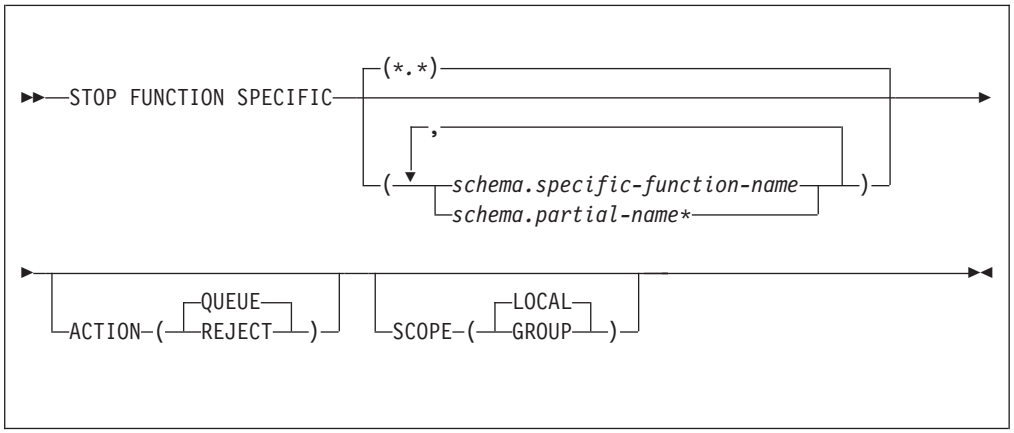
If you specify STOP FUNCTION SPECIFIC *.* or *schema.partial-name**, the privilege set of the process must include one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

(*.*)
Stops access to all functions, including functions that DB2 applications have not yet accessed.

If no functions are named, all functions are stopped.

schema.specific-function-name
Stops one specific function name. You cannot specify a function name as you can in SQL; you must use the specific name. If a specific name was not specified on the CREATE FUNCTION statement, query SYSIBM.SYSROUTINES for the correct specific name:

```
SELECT SPECIFICNAME, PARM_COUNT
  FROM SYSIBM.SYSROUTINES
   WHERE NAME='function_name'
    AND SCHEMA='schema_name';
```

For overloaded functions, this query can return multiple rows.

*schema.partial-name**
Stops a set of functions in the specified schema. The specific names of all functions in the set begin with *partial-name* and can end with any string, including the empty string. For example, schema1.ABC* stops all functions with specific names that begin with ABC in schema1.

ACTION
Indicates what to do with an SQL statement that invokes the function while the function is stopped. If you issue STOP FUNCTION SPECIFIC more than once for a given function, the action that is taken is determined by the ACTION option on the most recent command.

(QUEUE)
Queues the request until either of the following conditions is true:

- The wait exceeds the installation timeout value.
- You issue START FUNCTION SPECIFIC command for the function.

(REJECT)
Rejects the request.

SCOPE
Specifies the scope of the command.

(LOCAL)
Specify to stop the function on the local member only.

(GROUP)
Specify to stop the function on all members of the data sharing group.

Usage notes

Limitations of STOP FUNCTION SPECIFIC: STOP FUNCTION SPECIFIC is only applicable to external functions that run in the WLM application environment. STOP FUNCTION SPECIFIC cannot stop a built-in function or a user-defined function sourced on another function.

Permanently disabling a function: A stopped function does not remain stopped if DB2 is stopped and restarted. To disable a function permanently, you can:

- Use ALTER FUNCTION to change the LOADMOD name to a nonexistent z/OS load module
- Rename or delete the z/OS load module

Examples

Example 1: Stop access to all functions. While the STOP FUNCTION SPECIFIC command is in effect, DB2 queues all attempts to execute functions.

```
-STOP FUNCTION SPECIFIC ACTION(QUEUE)
```

This command produces output similar to the following output:

```
DSN9022I - DSNX9COM '-STOP FUNC' NORMAL COMPLETION
```

Example 2: Stop access to all functions. While the STOP FUNCTION SPECIFIC command is in effect, DB2 rejects attempts to execute functions.

```
-STOP FUNCTION SPECIFIC ACTION(REJECT)
```

This command produces output similar to the following output:

```
DSN9022I - DSNX9COM '-STOP FUNC' NORMAL COMPLETION
```

Example 3: Stop functions PAYROLL.USERFN1 and PAYROLL.USERFN3. While the STOP FUNCTION SPECIFIC command is in effect, DB2 queues all attempts to execute functions.

```
-STOP FUNCTION SPECIFIC(PAYROLL.USERFN1,PAYROLL.USERFN3)
```

This command produces output similar to the following output:

```
DSN9022I - DSNX9COM '-STOP FUNC' NORMAL COMPLETION
```

Example 4: Stop functions PAYROLL.USERFN1 and PAYROLL.USERFN3. While the STOP FUNCTION SPECIFIC command is in effect, DB2 rejects attempts to execute either of these functions.

```
-STOP FUNCTION SPECIFIC(PAYROLL.USERFN1,PAYROLL.USERFN3) ACTION(REJECT)
```

This command produces output similar to the following output:

```
DSN9022I - DSNX9COM '-STOP FUNC' NORMAL COMPLETION
```

Chapter 83. STOP irlmproc (z/OS IRLM)

The STOP *irlmproc* command shuts IRLM down normally. The command is rejected if any active DB2 subsystems are currently identified to IRLM.

Abbreviation: P

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option description"
- "Usage note"
- "Example" on page 414

Environment

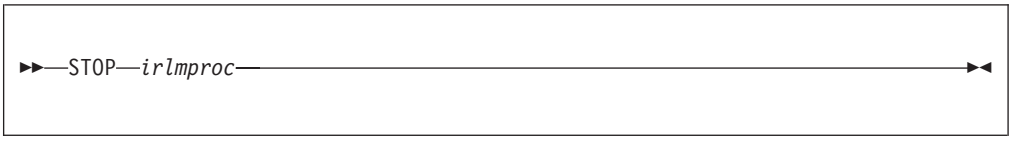
This command can be issued only from a z/OS console.

Data sharing scope: Member

Authorization

The command requires an appropriate level of operating system authority, as described in *z/OS MVS System Commands*.

Syntax



Option description

irlmproc
Identifies the procedure name for the IRLM to be stopped.

Usage note

Terminating the IRLM: If IRLM does not shut down normally, issue the MODIFY irlmproc,ABEND command to terminate the IRLM abnormally.

```
F irlmproc,ABEND,DUMP
```

If outstanding DB2 requests are in process and IRLM does not terminate, use the z/OS CANCEL command. If all other means of removing the subsystem fail, issue the z/OS FORCE CANCEL command.

© Copyright IBM Corp. 1983, 2008

Example

Enter the following command on the system console:

```
P KRLM1
```

IRLM outputs the following responses on system console:

```
DXR165I IR21 TERMINATED VIA IRLM MODIFY COMMAND
DXR121I IR21 END-OF-TASK CLEANUP SUCCESSFUL - HI-CSA     325K
```

In a data sharing environment: You cannot issue the STOP irlmproc command to IRLM in a data sharing group until no DB2 subsystems are identified to that IRLM and the IRLM issues the following messages:

```
DXR136I IR21 HAS DISCONNECTED FROM THE DATA SHARING GROUP
```

Any members that are still active in the group issue:

```
DXR137I JR21 GROUP STATUS CHANGED. IR21 233 HAS BEEN DISCONNECTED
FROM THE DATA SHARING GROUP
```

Chapter 84. -STOP PROCEDURE (DB2)

The DB2 command STOP PROCEDURE prevents DB2 from accepting SQL CALL statements for one or more stored procedures. You can qualify stored procedure names with a schema name. This command does not prevent CALL statements from running if they have already been queued or scheduled by DB2.

If the DB2 established stored procedure address space is connected to DB2, z/OS stops it based on the syntax of the STOP PROCEDURE command.

DB2 implicitly issues the command STOP PROCEDURE ACTION(REJECT) for any stored procedure that exceeds the maximum abend count. That count is set by the MAX ABEND COUNT field of installation panel DSNTIPX.

Abbreviation: -STO PROC

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 416
- "Option descriptions" on page 416
- "Usage notes" on page 417
- "Examples" on page 417

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group or local, depending on the value of the SCOPE option.

Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities:

- Ownership of the stored procedure
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

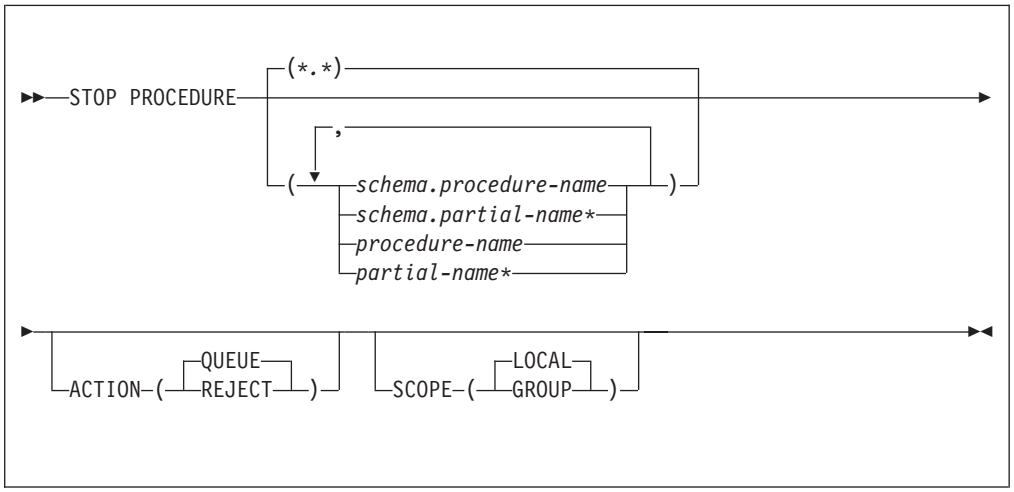
If you specify STOP PROCEDURE *.* or *schema.partial-name**, the privilege set of the process must include one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

Syntax



Option descriptions

(*.*)
Stops access to all stored procedures in all schemas, including procedure definitions that have not yet been accessed by DB2 applications. The DB2-established stored procedures address space terminates after active work is complete.

(*schema.procedure-name***)**
Identifies the fully-qualified procedure name that is to be stopped.

(*schema.partial-name****)**
Stops a set of stored procedures in the specified schema. The names of all procedures in the set begin with *partial-name* and can end with any string, including the empty string. For example, PAYROLL.* stops all stored procedures in the PAYROLL schema.

procedure-name
Identifies one or more specific stored procedure names to be stopped. The procedure name is implicitly qualified with the SYSPROC schema name.

*partial-name**
Stops a set of stored procedures within the SYSPROC schema. The names of all procedures in the set begin with *partial-name* and can end with any string, including the empty string. For example, ABC* stops all stored procedures with names that begin with ABC.

ACTION
Indicates what to do with a CALL statement that is received while the procedure is stopped. If STOP PROCEDURE is issued more than once for a given procedure, the action taken is determined by the ACTION option on the most recent command.

(QUEUE)
Queues the request until either:

- The wait exceeds the installation timeout value, or
- The stored procedure is started by the command START PROCEDURE.

(REJECT)
Rejects the request

SCOPE
Specifies the scope of the command.

(LOCAL)
Specify to stop the procedure on the local member only.

(GROUP)
Specify to stop the procedure on all members of the data sharing group.

Usage notes

Permanently disabling a stored procedure: A stopped procedure does not remain stopped if DB2 is stopped and restarted. To disable a stored procedure permanently, you can:

- Drop the procedure using the DROP PROCEDURE statement. See Chapter 5 of *DB2 SQL Reference* for more information.
- Use an ALTER PROCEDURE statement.
- Rename or delete the z/OS load module.

Stored procedure address space management differences: The STOP PROCEDURE command operates differently depending on how the DB2 stored procedures address spaces are established. For more information, see Part 4 (Volume 1) of *DB2 Administration Guide*.

Examples

Example 1: Stop access to all stored procedures, and terminate the DB2 stored procedures address space. While the STOP PROCEDURE command is in effect, attempts to execute stored procedures are queued.

```
-STOP PROCEDURE ACTION(QUEUE)
```

```
DSNX947I - DSNX9SP2 STOP PROCEDURE SUCCESSFUL FOR *.*
DSN9022I - DSNX9COM '-STOP PROC' NORMAL COMPLETION
```

Example 2: Stop access to all stored procedures, and terminate the DB2 stored procedures address space. While the STOP PROCEDURE command is in effect, attempts to execute stored procedures are rejected.

```
-STOP PROCEDURE ACTION(REJECT)
```

```
DSNX947I - DSNX9SP2 STOP PROCEDURE SUCCESSFUL FOR *.*
DSN9022I - DSNX9COM '-STOP PROC' NORMAL COMPLETION
```

Example 3: Stop stored procedures USERPRC1 and USERPRC3. While the STOP PROCEDURE command is in effect, attempts to execute these stored procedure are queued.

```
-STOP PROCEDURE(USERPRC1,USERPRC3)
```

```
DSNX947I - DSNX9SP2 STOP PROCEDURE SUCCESSFUL FOR USERPRC1
DSNX947I - DSNX9SP2 STOP PROCEDURE SUCCESSFUL FOR USERPRC3
DSN9022I - DSNX9COM '-STOP PROC' NORMAL COMPLETION
```

Example 4: Stop stored procedures USERPRC1 and USERPRC3. While the STOP PROCEDURE command is in effect, attempts to execute these stored procedure are rejected.

```
-STOP PROCEDURE(USERPRC1,USERPRC3) ACTION(REJECT)
```

```
DSNX947I - DSNX9SP2 STOP PROCEDURE SUCCESSFUL FOR USERPRC1
DSNX947I - DSNX9SP2 STOP PROCEDURE SUCCESSFUL FOR USERPRC3
DSN9022I - DSNX9COM '-STOP PROC' NORMAL COMPLETION
```

Chapter 85. -STOP RLIMIT (DB2)

The DB2 command STOP RLIMIT stops the resource limit facility. STOP RLIMIT resets all previously set limits to infinity and resets the accumulated time to zero. All previously limited SQL statements (SELECT, UPDATE, DELETE, and INSERT) executed through an SQL PREPARE or EXECUTE IMMEDIATE statement run with no limit.

Abbreviation: -STO RLIM

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Example"

Environment

This command can be issued from a z/OS console, a DSN session under TSO, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Member

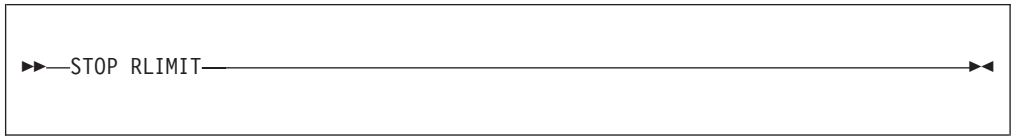
Authorization

To execute this command, you must use a privilege set of the process that includes one of the following authorities:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

Syntax



Example

Stop the resource limit facility.

```
-STOP RLIMIT
```

© Copyright IBM Corp. 1983, 2008

Chapter 86. -STOP TRACE (DB2)

The DB2 command STOP TRACE stops tracing.

One additional option to this command and additional values for a few other options exist. This additional information is intended for service and use under the direction of IBM Software Support. For details, see *DB2 Diagnosis Guide and Reference*.

Abbreviation: -STO TRA

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 422
- "Option descriptions" on page 423
- "Usage notes" on page 425
- "Examples" on page 426

Environment

This command can be issued from a z/OS console, a DSN session, a DB2I panel (DB2 COMMANDS), an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group or local, depending on the value of the SCOPE option.

Authorization

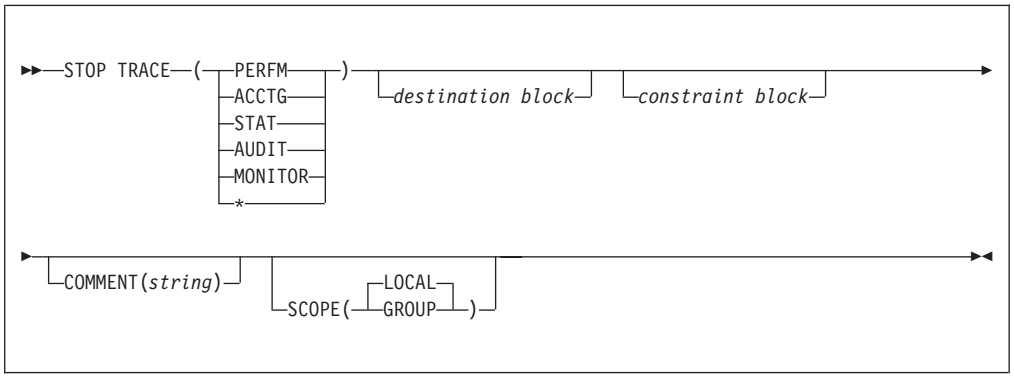
To execute this command, you must use a privilege set of the process that includes one of the following privileges or authorities:

- TRACE privilege
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

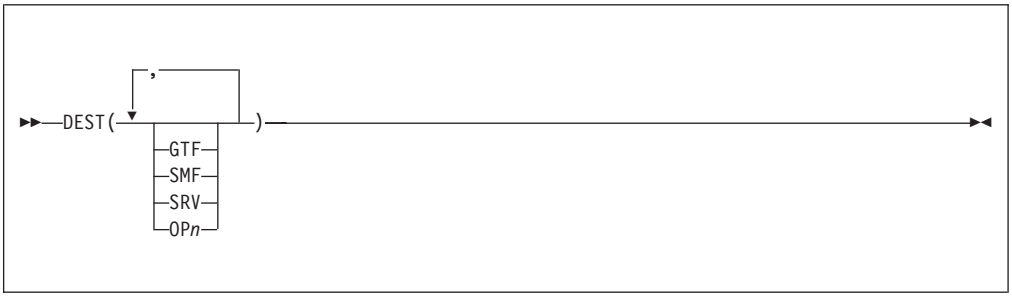
DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

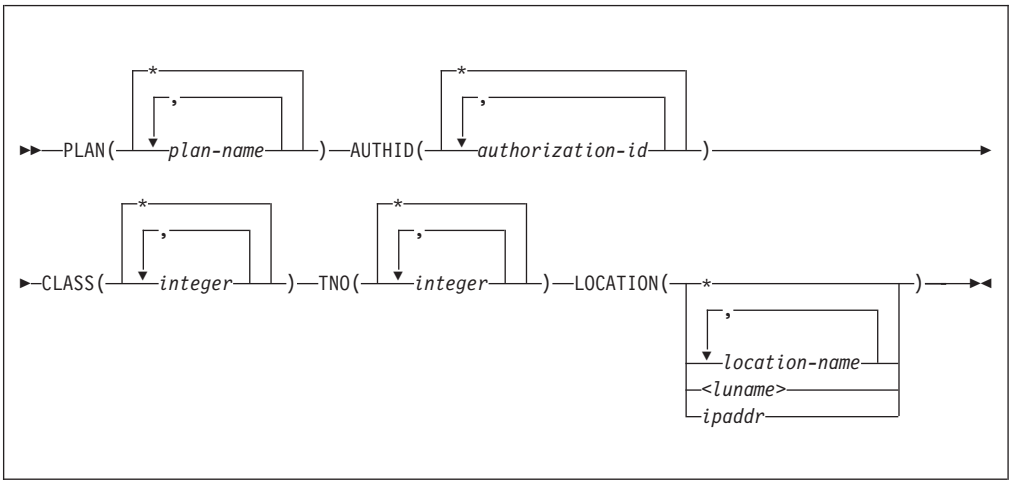
Syntax


STOP TRACE
(
PERFM
ACCTG
STAT
AUDIT
MONITOR
*
)
destination block
constraint block
COMMENT(string)
SCOPE(
LOCAL
GROUP
)

destination block:



constraint block:



Option descriptions

For additional descriptions of each of the following trace types, see Chapter 77, "-START TRACE (DB2)," on page 383.

(PERFM)
Specify to stop a trace that is intended for performance analysis and tuning.

Abbreviation: P

(ACCTG)
Specify to stop an accounting trace.

Abbreviation: A

(STAT)
Specify to stop a trace that collects statistical data. The LOCATION option cannot be specified when you choose a statistics trace.

Abbreviation: S

(AUDIT)
Specify to stop a trace that collects audit data from various components of DB2.

Abbreviation: AU

(MONITOR)
Specify to stop a trace that collects monitor data.

Abbreviation: MON

(*) Specify to stop all trace activity. See "Usage notes" on page 425 for information about using STOP TRACE (*) with traces that use monitor trace class 6.

SCOPE
Specifies the scope of the command.

(LOCAL)
Stops the trace only on the local DB2 subsystem.

(GROUP)
Stops the trace on all members of a data sharing group.

COMMENT(*string***)**
Gives a comment that is reproduced in the trace output record for the STOP TRACE command (except in the resident trace tables).

string is any SQL string; it must be enclosed between apostrophes if it includes a blank, comma, or special character.

DEST
Limits stopping to traces started for particular destinations. You can use more than one value, but do not use the same value twice. If you do not specify a value for DEST, DB2 does not use destination to limit which traces to stop.

Abbreviation: D

Possible values and their meanings are:

Value	Trace destination
GTF	The generalized trace facility
SMF	The System Management Facility
SRV	An exit to a user-written routine
OP*n*	A specific destination. *n* can be a value from 1 to 8

See Chapter 77, "-START TRACE (DB2)," on page 383 for a list of allowable destinations for each trace type.

PLAN(*plan-name***, ...)**
Limits stopping to traces started for particular application plans. You can use up to eight plan names. If you use more than one name, you can use only one value for AUTHID, TNO, and LOCATION. Do not use this option with STAT.

The **default** is **PLAN(*)**, which does not limit the command.

AUTHID(*authorization-id***, ...)**
Limits stopping to traces started for particular authorization identifiers. You can use up to eight identifiers. If you use more than one identifier, you can use only one value for PLAN, TNO, and LOCATION. Do not use this option with STAT.

The **default** is **AUTHID(*)**, which does not limit the command.

CLASS(*integer***, ...)**
Limits stopping to traces started for particular classes. For descriptions of the allowable classes, see Chapter 77, "-START TRACE (DB2)," on page 383. You cannot specify a class if you did not specify a trace type.

Abbreviation: C

The **default** is **CLASS(*)**, which does not limit the command.

TNO(*integer***, ...)**
Limits stopping to particular traces, identified by their trace numbers (1 to 32,

01 to 09). You can use up to eight trace numbers. If you use more than one number, you can use only one value each for PLAN, AUTHID, and LOCATION.

The **default** is **TNO(*)**, which does not limit the command.

LOCATION(*location-name***, ...)**
Introduces a list of specific location names for which traces are stopped. Limits the traces you can stop to those started for threads with connections to remote locations; the use of the LOCATION option precludes stopping traces of non-distributed threads.

You can specify up to eight location names. If you use more than one location name, you can only use one value for PLAN, AUTHID, and TNO. You cannot use this option with STAT.

The **default** is **LOCATION( )**, which does not limit the command.

(*) LOCATION(*) limits the command to those traces that were started with the one or more location names specified on the LOCATION keyword of START TRACE.

<*luname***>**
Stops the DB2 trace for the remote clients that are connected to DDF through the remote SNA LU that you specify in <*luname*>.

ipaddr
Stops the DB2 trace for remote clients that are connected to DDF through the remote TCP/IP host. *nnn.nnn.nnn.nnn*is the dotted decimal IP address.

Requesters other than DB2 UDB for z/OS: DB2 does not receive a location name from requesters that are not DB2. To display information about a requester that is not a DB2 UDB for z/OS subsystem, enter its LUNAME, enclosed by the less-than (<) and greater-than (>) symbols. For example, to display information about a requester with the LUNAME of LULA, enter the following command:

```
-STOP TRACE (*) LOCATION (<LULA>)
```

DB2 uses the <LUNAME> notation in messages displaying information about requesters that are not DB2 UDB for z/OS.

Usage notes

Stopping specific traces: Each option that you use, except TNO, limits the effect of the command to active traces that were started using the same option, either explicitly or by default, with exactly the same parameter values. For example, the following command stops only the active traces that were started using the options PERFM *and* CLASS (1,2):

```
-STOP TRACE (PERFM) CLASS (1,2)
```

This command does *not* stop, for example, any trace started using CLASS(1).

You must specify a trace type or an asterisk. For example, the following command stops all active traces:

```
-STOP TRACE (*)
```

Traces that use monitor trace class 6: When stopping trace classes, a special circumstance occurs if monitor trace class 6 is active. Monitor trace class 6 enables

and disables data propagation. To avoid accidentally stopping this trace class, the commands STOP TRACE(*) and STOP TRACE(MON) CLASS(*) fail if monitor trace class 6 is active.

To stop monitor trace class 6, you must explicitly specify it as one of the arguments of the CLASS option of the STOP TRACE command, including any other monitor trace classes that were started with monitor trace class 6. For example, if monitor trace class 6 was started with the command START TRACE(MON) CLASS(1,3,6), the following command stops it:

```
-STOP TRACE(MON) CLASS(1,3,6)
```

In the case where monitor trace class 6 was started with the command START TRACE(MON) CLASS(*), you must explicitly specify all 32 monitor trace classes to have monitor trace class 6 stopped:

```
-STOP TRACE(MON) CLASS(1,2,3,4,5,6,...32)
```

However, if monitor trace class 6 is not active the STOP TRACE(*) command stops all active traces.

Traces started by a IFI/IFC program: Before you stop an active trace, ensure that an IFI application program or the IFC Selective Dump utility (DSN1SDMP) did not start the trace. If you stop a trace started by DSN1SDMP, the DSN1SDMP utility abnormally terminates.

Examples

Example 1: Stop all traces that have the generalized trace facility as their only destination.

```
-STOP TRACE (*) DEST (GTF)
```

Example 2: Stop an accounting trace of all threads between the local and USIBMSTODB21 DB2 subsystems for plan DSN8BC81. Include a comment.

```
-STOP TRACE (ACCTG)
  PLAN (DSN8BC81)
  LOCATION (USIBMSTODB21)
  COMMENT('ACCTG TRACE FOR DSN8BC81')
```

Example 3: Stop trace number 4.

```
-STOP TRACE (P) TNO(4)
```

Example 4: Stop all active traces of any type for USIBMSTODB22.

```
-STOP TRACE (*) LOCATION (USIBMSTODB22)
```

Example 5: Stop all performance traces.

```
-STOP TRACE=P
```

Example 6: Stop all monitor tracing.

```
-STOP TRACE(MON)
```

Example 7: Stop all monitor tracing in a data sharing group.

```
-STOP TRACE(MON) SCOPE(GROUP)
```

Chapter 87. -TERM UTILITY (DB2)

The DB2 command TERM UTILITY terminates execution of a DB2 utility job step and releases all resources associated with the step. When executing, a utility does not terminate until it checks to see that the TERM UTILITY command was issued. Active utilities perform this check periodically. If the utility is stopped, all its resources are released by the TERM UTILITY command. An active utility can be terminated only from the DB2 on which it is running. A stopped utility can be terminated from any active member of the data sharing group.

Abbreviation: -TER UTIL

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax" on page 428
- "Option descriptions" on page 428
- "Usage notes" on page 428
- "Examples" on page 429

Environment

This command can be issued from a z/OS console, a DSN session, DB2I panels DB2 COMMANDS and DB2 UTILITIES, an IMS or CICS terminal, or a program using the instrumentation facility interface (IFI).

Data sharing scope: Group or member. The utility is implicitly of group scope when the utility is stopped. For release dependency information, see *DB2 Data Sharing: Planning and Administration*.

Authorization

To execute this command, you must use the primary or some secondary authorization ID of the process that originally submitted the utility job, or you must use a privilege set of the process that includes one of the following authorities:

- DBMAINT authority
- DBCTRL authority
- DBADM authority
- SYSOPR authority
- SYSCTRL authority
- SYSADM authority

The utilities DIAGNOSE, REPORT, and STOSPACE can be terminated only by the job submitter or by a holder of SYSOPR, SYSCTRL, or SYSADM authority.

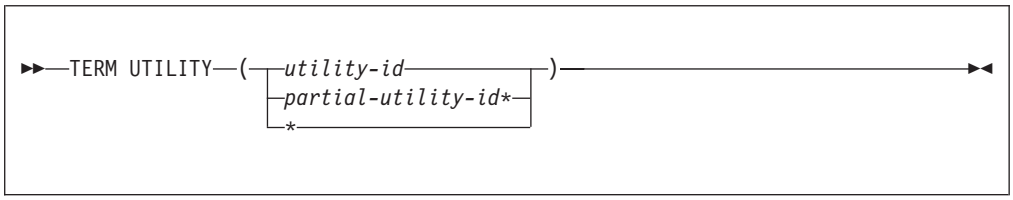
DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

© Copyright IBM Corp. 1983, 2008

For users with DBMAINT, DBCTRL, or DBADM authority, the command takes effect only when DB2 can determine that the user has sufficient authority over each object that the utility job accesses.

Database DSNDB06 contains the table spaces and index spaces that are required to check authorization. If a table or index space that is required for authorization checking is affected by a utility that you need to terminate, installation SYSADM authority is required to terminate that utility.

Syntax



Option descriptions

One of the following parameters must be specified.

(*utility-id***)**
Is the utility identifier, or the UID parameter used when creating the utility job step.

If *utility-id* was created by the DSNU CLIST by default, it has the form *tso-userid.control-file-name*. For the control file name that is associated with each utility, see the description of the DSNU utility in *DB2 Utility Guide and Reference*.

If *utility-id* was created by default by the EXEC statement invoking DSNUTILB, then the token has the form *userid.jobname*.

For lower case *utility-id*, enclose the entire name between apostrophes. For example, specify 'SMITH.'

(*partial-utility-id****)**
Terminates every utility job that begins with *partial-utility-id*. For example, TERM UTILITY(ABCD*) terminates every utility job step whose utility identifier begins with the letters ABCD. If you have a two-part utility ID, such as ABCD.EFGH, TERM UTILITY(ABCD*) also terminates that utility.

(*) Terminates every utility job step known to DB2 for which you are authorized.

Usage notes

Restarting utilities: A terminated utility job step cannot be restarted. You must resubmit the step as a new utility job.

What happens to particular utilities: In some cases, terminating a utility job can leave work in an undesirable state, requiring special processing before the job can be resubmitted. The following list describes the effects of TERM UTILITY on jobs for each of the utilities:

Utility	Special effects of the TERM UTILITY command
CATENFM	None.

CATMAINT	Places indexes in REBUILD-pending status.
CHECK DATA	Table spaces remain in CHECK-pending status.
CHECK INDEX	None.
CHECK LOB	Places LOB table spaces and indexes in the utility read-only (UTRO) state.
COPY	Inserts "T" record in SYSIBM.SYSCOPY. When you run COPY, COPY does not allow an incremental image copy if the "T" record exists.
DIAGNOSE	None.
LOAD	See *DB2 Utility Guide and Reference* for the effect of TERM on the LOAD utility phases.
MERGECOPY	None.
MODIFY RECOVERY	None.
MODIFY STATISTICS	None.
QUIESCE	None.
REBUILD INDEX	Places the object that is being rebuilt in REBUILD-pending status.
RECOVER	Places the object that is being recovered in RECOVER-pending status.
REORG INDEX	See *DB2 Utility Guide and Reference* for the effect of TERM on the REORG INDEX utility phases.
REORG TABLESPACE	See *DB2 Utility Guide and Reference* for the effect of TERM on the REORG TABLESPACE utility phases.
REPAIR	None.
REPORT	None.
RUNSTATS	None.
STOSPACE	None.
UNLOAD	The output data set remains incomplete until you restart the utility job or delete the data set.

Examples

Example 1: Terminate all utility jobs for which you are authorized.

```
-TERM UTILITY (*)
```

Example 2: Terminate all utility jobs whose utility ID begins with SMITH.

```
-TERM UTILITY
  (SMITH*)
```

Example 3: Terminate utility job whose utility ID is in lowercase characters.

```
-TERM UTILITY
  ('SMITH')
```

Chapter 88. /TRACE (IMS)

The IMS /TRACE command directs and controls the IMS capabilities for tracing internal IMS events. It also starts, stops, and defines the activity to be monitored by the IMS Monitor. For more information about the IMS Monitor, see *IMS Utilities Reference: System*.

Abbreviation: /TRA

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions"
- "Examples" on page 432

Environment

This command can be issued only from an IMS terminal.

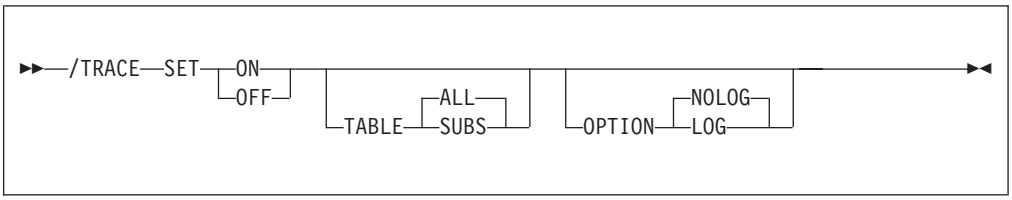
Data sharing scope: Member

Authorization

To enter this command, users must have passed the IMS security check, as described in *IMS Administration Guide: System*.

The syntax diagram includes only those parameters that DB2 users need to know. For a diagram with the complete syntax of this command, see *IMS Command Reference*.

Syntax



Option descriptions

The option descriptions for the /TRACE command are described in *IMS Command Reference*; however, this section provides information about the two parameters that are especially important for DB2 users.

SUBS
Indicates that the external subsystem trace table (containing information about every interaction with DB2) is to be enabled or disabled. SET ON TABLE SUBS enables the DB2 trace facility, and SET OFF TABLE SUBS disables it.

© Copyright IBM Corp. 1983, 2008

If nothing is specified with the TABLE keyword, then the default is ALL; ALL includes SUBS, as well as other trace tables.

LOG
Specifies that traced data is to be written to the IMS system log. Because IMS has a tracing mechanism that writes trace entries to the IMS system log, it is important that DB2 users specify SET ON and TABLE OPTION LOG. Otherwise, the trace information that IMS provides will not be available unless a control region dump occurs.

Examples

Example 1: This command starts IMS tracing and:

- Enables the DB2 trace
- Writes IMS trace tables to the IMS log before they wrap.

```
/TRACE SET ON TABLE SUBS OPTION LOG
```

Example 2: This command starts IMS tracing and:

- Enables all trace tables (including DB2 trace tables); (ALL is the default parameter for the TABLE keyword)
- Writes IMS trace tables to the IMS log before they wrap.

```
/TRACE SET ON TABLE ALL OPTION LOG
```

Chapter 89. TRACE CT (z/OS IRLM)

The z/OS command TRACE CT starts, stops, or modifies a diagnostic trace for the internal resource lock manager (IRLM) of DB2. IRLM does not support all the options available on the TRACE command as described in *z/OS MVS System Commands*.

The following topics provide additional information:

- "Environment"
- "Authorization"
- "Syntax"
- "Option descriptions" on page 434
- "Usage notes" on page 435

Environment

This command can be issued only from a z/OS console.

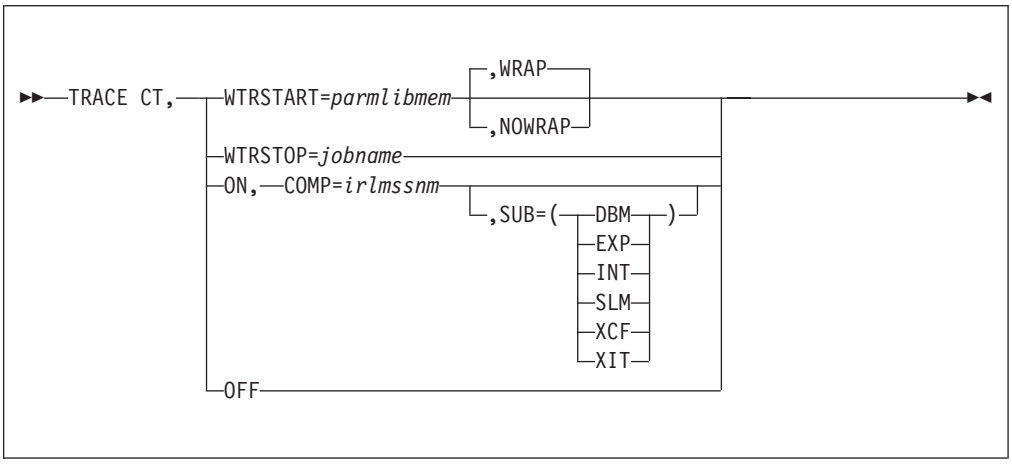
Data sharing scope: Member

Authorization

This command requires an appropriate level of operating system authority, as described in *z/OS MVS System Commands*.

The syntax diagram and option descriptions for this command are purposely incomplete. Options that are not shown are described in *z/OS MVS System Commands*.

Syntax



© Copyright IBM Corp. 1983, 2008

Option descriptions

CT

Specifies the component trace. (Do not use other trace options available on the z/OS TRACE command).

WTRSTART=*parmlibmem*

Identifies the member that contains source JCL. That JCL executes the CTRACE writer and defines the data set to which it writes the trace buffers. This member can be a procedure cataloged in SYS1.PROCLIB or a job.

WRAP

Specifies that when the system reaches the end of the group of data sets, it writes over the oldest data at the beginning of the first data set in the group. The system uses only the primary extents of the data sets.

NOWRAP

Specifies that the system stops writing to the data sets when they are all full. The system uses the primary and secondary extents of the data sets.

WTRSTOP=*jobname*

Stops the CTRACE writer for a trace that is running. The system also closes the data sets that the writer used.

jobname identifies the trace, either by:

- Member name, if the source JCL is a procedure
- Job name, if that appears on a JOB statement in the source JCL

ON

Turns on the trace.

COMP=*irlmssnm*

Gives the IRLM subsystem name.

SUB=*subname*

Specifies the type of sublevel trace. Traces INT, EXP, and XIT are ON by default. You cannot turn off traces INT and EXP. If you do not specify a subname on the TRACE command, the trace is performed on all subnames that you control. Specifying one subname restricts the traces to that trace plus the EXP and INT traces.

Use:	To trace:
DBM	Interactions with the identified DBMS
EXP	Any exception condition
INT	Member and group events outside normal locking activity
SLM	Interactions with the z/OS locking component
XCF	All interactions with z/OS cross-system coupling services
XIT	Only asynchronous interactions with the z/OS locking component

OFF

Turns off the trace. If IRLM is connected to a CTRACE writer for the component trace, the system disconnects it.

Usage notes

Include the IRLM load module in the z/OS link list: This command uses z/OS component trace services. Include the IRLM load module DXRRL183, which contains a routine for stopping and starting, in the z/OS link list.

Displaying a trace: To display a trace, use the z/OS DISPLAY command:

```
D TRACE,COMP=IRLM
```

The z/OS DISPLAY TRACE command output is incorrect for IRLM unless you use TRACE CT commands to inform z/OS of the TRACE status. IRLM initializes it's own traces and writes them in CTRACE format, but IRLM has no interface to z/OS to inform it of the status. If you want to know the true status of the traces without using TRACE CT commands to inform z/OS, use the MODIFY *irlmproc*,STATUS,TRACE command.

Monitoring a trace: To monitor a trace, use the z/OS MODIFY *irlmproc*,STATUS,TRACE command, on page 287.

Setting the number of trace buffers: To set the number of trace buffers used by traces, use the z/OS MODIFY *irlmproc*,SET command, on page 283.

Sample procedure for the CTRACE writer: This procedure identifies the data set to which the next sample procedure writes data. The external trace writer must be executed at the same or higher dispatch priority as IRLM. This allows the I/O to keep up with the filling of the trace buffers.

```
//CTWTR    PROC
//         EXEC PGM=ITTTRCWR
//TRCOUT01 DD   DSNAME=SYS1.WTR1,DISP=OLD
//TRCOUT02 DD   DSNAME=SYS1.WTR2,DISP=OLD
```

Sample procedure to start and stop a DBM trace to the CTRACE writer: After you enter the command TRACE CT,WTRSTART, turn the trace on and connect the writer, using the WTR parameter in the reply for the command TRACE CT.

```
TRACE CT,WTRSTART=CTWTR
TRACE CT,ON,COMP=IRLM,SUB=(DBM)
⋮
  (z/OS asks for a reply)
⋮
R 15,WTR=CTWTR,END
TRACE CT,OFF,COMP=IRLM,SUB=(DBM)
⋮
  (Wait to make sure trace buffers are externalized.)
TRACE CT,WTRSTOP=CTWTR
```

Sample procedure to start and stop traces in wrap-around mode: Traces captured in this procedure are saved in a limited number of buffers that are provided by IRLM. Each buffer is reused when the previous buffer is filled. To start the trace in this wrap-around mode, enter the following commands:

```
TRACE CT,ON,COMP=IRLM
⋮
  (z/OS asks for a reply)
⋮
R 15,END
⋮
TRACE CT,OFF,COMP=IRLM
```

Impact of setting TRACE CT ON: Each active subname type requires up to 0.7 MB of ECSA. Because IRLM initializes its own traces when it starts, the DISPLAY TRACE command shows that all traces are off. After you issue the TRACE ON command, the reports are accurate except for the two subname types, INT and EXT, which cannot be turned off.

Part 4. Appendixes

© Copyright IBM Corp. 1983, 2008

Appendix. Directory of subsystem parameters

This appendix describes the subsystem parameters.

The following topics provide additional information:

- "Editing the subsystem parameters and DSNHDECP values"
- "Directory of subsystem parameters and DSNHDECP values"

Editing the subsystem parameters and DSNHDECP values

The subsystem parameter module is generated by job DSNTIJUZ each time you install, migrate, or update DB2. Seven macros expand to form this data-only subsystem parameter load module. It contains the DB2 execution-time parameters that you selected using the ISPF panels. These seven macros are DSN6ARVP, DSN6ENV, DSN6FAC, DSN6LOGP, DSN6SPRM, DSN6SYSP, and DSN6GRP.

The data-only load module DSNHDECP is also generated by job DSNTIJUZ. It contains the application programming defaults.

Directory of subsystem parameters and DSNHDECP values

Table 37 shows you each macro parameter, the macro where it is located, installation panel name, whether it can be updated online. Online update capability does not apply to macro DSNHDECP so values are not listed for those parameters.

Some parameters, when updated online, result in a change in system behavior. These parameters include:

- PARTKEYU
- SYSADM/SYSADM2
- CACHEDYN
- MAXKEEPD
- XLKUPDLT

Table 37. Directory of subsystem parameters and DSNHDECP values

Parameter	Macro	Panel	Update Online
ABEXP	DSN6SPRM	DSNTIPO	Yes
ABIND	DSN6SPRM	DSNTIPO	Yes
ACCUMACC	DSN6SYSP	DSNTIPN	Yes
ACCUMUID	DSN6SYSP	DSNTIPN	Yes
AEXITLIM	DSN6SPRM	DSNTIPP	Yes
AGCCSID	DSNHDECP	DSNTIPF	—
ALCUNIT	DSN6ARVP	DSNTIPA	Yes
ALL/dbname	DSN6SPRM	DSNTIPS	No
AMCCSID	DSNHDECP	DSNTIPF	—
APPENSCH	DSNHDECP	DSNTIPF	—
ARCPFX1	DSN6ARVP	DSNTIPH	Yes
ARCPFX2	DSN6ARVP	DSNTIPH	Yes
ARCRETN	DSN6ARVP	DSNTIPA	Yes
ARCWRTC	DSN6ARVP	DSNTIPA	Yes

© Copyright IBM Corp. 1983, 2008

Table 37. Directory of subsystem parameters and DSNHDECP values (continued)

Parameter	Macro	Panel	Update Online
ARCWTOR	DSN6ARVP	DSNTIPA	Yes
ARC2FRST	DSN6LOGP	DSNTIPO	Yes
ASCCSID	DSNHDECP	DSNTIPF	—
ASSIST	DSN6GRP	DSNTIPK	No
AUDITST	DSN6SYSP	DSNTIPN	No
AUTH	DSN6SPRM	DSNTIPP	No
AUTHCACH	DSN6SPRM	DSNTIPP	Yes
BACKODUR	DSN6SYSP	DSNTIPL	No
BINDNV	DSN6SPRM	DSNTIPP	Yes
BLKSIZE	DSN6ARVP	DSNTIPA	Yes
BMPTOUT	DSN6SPRM	DSNTIPI	Yes
CACHEDYN	DSN6SPRM	DSNTIP8	Yes
CACHEDYN_FREELOCAL	DSN6SPRM	—	Yes
CACHEPAC	DSN6SPRM	DSNTIPP	No
CACHERAC	DSN6SPRM	DSNTIPP	No
CATALOG	DSN6ARVP	DSNTIPA	Yes
"	DSN6SPRM	DSNTIPA2	—
CDSSRDEF	DSN6SPRM	DSNTIP8	Yes
CHARSET	DSNHDECP	DSNTIPF	—
CHGDC	DSN6SPRM	DSNTIPO	Yes
CHKFREQ	DSN6SYSP	DSNTIPL	Yes
CMTSTAT	DSN6FAC	DSNTIPR	No
COMCRIT	DSN6SPRM	—	Yes
COMPACT	DSN6ARVP	DSNTIPA	Yes
COMPAT	DSNHDECP	—	—
CONDBAT	DSN6SYSP	DSNTIPE	Yes
CONTSTOR	DSN6SPRM	DSNTIPE	Yes
COORDNTR	DSN6GRP	DSNTIPK	No
CTHREAD	DSN6SYSP	DSNTIPE	Yes
DBACRVW	DSN6SPRM	DSNTIPP	Yes
DBPROTCL	DSN6SYSP	DSNTIP5	Yes
DATE	DSNHDECP	DSNTIP4	—
DATELEN	DSNHDECP	DSNTIP4	—
DB2SUPLD	DSNHDECP	—	—
DDF	DSN6FAC	DSNTIPR	No
DEALLCT	DSN6LOGP	DSNTIPA	Yes
DECARTH	DSNHDECP	DSNTIP4	—
DECDIV3	DSN6SPRM	DSNTIPF	No
DECIMAL	DSNHDECP	DSNTIPF	—
DEFLANG	DSNHDECP	DSNTIPF	—
DEFLTID	DSN6SPRM	DSNTIPP	No
DELIM	DSNHDECP	DSNTIPF	—
DESCSTAT	DSN6SPRM	DSNTIP4	Yes
DLDFREQ	DSN6SYSP	DSNTIPL	Yes
DLITOUT	DSN6SPRM	DSNTIPI	Yes
DSHARE	DSN6GRP	DSNTIPA1	No
DSMAX	DSN6SPRM	DSNTIPC	Yes
DSQLDELI	DSNHDECP	DSNTIPF	—
DSSTIME	DSN6SYSP	DSNTIPN	Yes
DSVCI	DSN6SYSP	DSNTIP7	Yes
DYNRULES	DSNHDECP	DSNTIP4	—
EDMBFIT	DSN6SPRM	DSNTIP8	Yes

Table 37. Directory of subsystem parameters and DSNHDECP values (continued)

	Parameter	Macro	Panel	Update Online
	EDMDBDC	DSN6SPRM	DSNTIPC	Yes
	EDMSTMTC	DSN6SPRM	DSNTIPC	Yes
	EDMPOOL	DSN6SPRM	DSNTIPC	Yes
	EDPROP	DSN6SPRM	DSNTIPO	Yes
	ENSCHEME	DSNHDECP	DSNTIPF	—
	EVALUNC	DSN6SPRM	DSNTIP8	Yes
	EXTRAREQ	DSN6SYSP	DSNTIP5	Yes
	EXTRASRV	DSN6SYSP	DSNTIP5	Yes
	EXTSEC	DSN6SYSP	DSNTIPR	Yes
	GCCSID	DSNHDECP	DSNTIPF	—
	GRPNAME	DSN6GRP	DSNTIPK	No
	HOPAUTH	DSN6SPRM	DSNTIP5	No
	IDBACK	DSN6SYSP	DSNTIPE	Yes
	IDFORE	DSN6SYSP	DSNTIPE	Yes
	IDTHTOIN	DSN6FAC	DSNTIPR	Yes
	IDXBPOOL	DSN6SYSP	DSNTIP1	Yes
#	IGNSORTN	DSN6SPRM	—	Yes
	IMMEDWRI	DSN6GRP	DSNTIP8	Yes
	INLISTP	DSN6SPRM	—	Yes
	IRLMAUT	DSN6SPRM	DSNTIPI	No
	IRLMPRC	DSN6SPRM	DSNTIPI	No
	IRLMRWT	DSN6SPRM	DSNTIPI	No
	IRLMSID	DSN6SPRM	DSNTIPI	No
	IRLMSWT	DSN6SPRM	DSNTIPI	Yes
	IXQTY	DSN6SYSP	DSNTIP7	Yes
	LBACKOUT	DSN6SYSP	DSNTIPL	No
	LC_CTYPE	DSNHDECP	DSNTIPF	—
	LEMAX	DSN6SPRM	DSNTIP7	No
	LOBVALA	DSN6SYSP	DSNTIP7	Yes
	LOBVALS	DSN6SYSP	DSNTIP7	Yes
	LOGAPSTG	DSN6SYSP	DSNTIPL	No
	LRDRTHLD	DSN6SYSP	DSNTIPE	Yes
	MAINTYPE	DSN6SPRM	DSNTIP8	Yes
	MAX_NUM_CUR	DSN6SPRM	DSNTIPX	Yes
	MAX_ST_PROC	DSN6SPRM	DSNTIPX	Yes
#	MAX_UTIL_PARTS	DSN6SPRM	—	Yes
	MAXARCH	DSN6LOGP	DSNTIPA	No
	MAXDBAT	DSN6SYSP	DSNTIPE	Yes
	MAXKEEPD	DSN6SPRM	DSNTIPE	Yes
	MAXRBLK	DSN6SPRM	DSNTIPC	Yes
	MAXRTU	DSN6LOGP	DSNTIPA	Yes
	MAXTYPE1	DSN6FAC	DSNTIPR	Yes
	MCCSID	DSNHDECP	DSNTIPF	—
	MEMBNAME	DSN6GRP	DSNTIPK	No
	MGEXTSZ	DSN6SYSP	DSNTIP7	Yes
	MINSTOR	DSN6SPRM	DSNTIPE	Yes
	MIXED	DSNHDECP	DSNTIPF	—
	MON	DSN6SYSP	DSNTIPN	No
	MONSIZE	DSN6SYSP	DSNTIPN	No
#	MORE_UNION_DISTRIBUTION	DSN6SPRM	—	Yes
	NEWFUN	DSNHDECP	DSNTIPA1	—
	NPGTHRSH	DSN6SPRM	—	Yes

Table 37. Directory of subsystem parameters and DSNHDECP values (continued)

Parameter	Macro	Panel	Update Online
NUMLKTS	DSN6SPRM	DSNTIPJ	Yes
NUMLKUS	DSN6SPRM	DSNTIPJ	Yes
# OJPERFEH	DSN6SPRM	—	No
OPTHINTS	DSN6SPRM	DSNTIP8	Yes
OUTBUFF	DSN6LOGP	DSNTIPL	No
PADIX	DSN6SPRM	DSNTIPE	Yes
PADNTSTR	DSNHDECP	DSNTIP4	No
PARAMDEG	DSN6SPRM	DSNTIP8	Yes
PARTKEYU	DSN6SPRM	DSNTIP8	Yes
PCLOSEN	DSN6SYSP	DSNTIPL	Yes
PCLOSET	DSN6SYSP	DSNTIPL	Yes
POOLINAC	DSN6FAC	DSNTIP5	Yes
PRIQTY	DSN6ARVP	DSNTIPA	Yes
PROTECT	DSN6ARVP	DSNTIPP	Yes
PTASKROL	DSN6SYSP	—	Yes
QUIESCE	DSN6ARVP	DSNTIPA	Yes
RECALL	DSN6SPRM	DSNTIPO	No
RECALLD	DSN6SPRM	DSNTIPO	Yes
REFSHAGE	DSN6SPRM	DSNTIP8	Yes
RELCURHL	DSN6SPRM	DSNTIP8	Yes
# RESTART/DEFER	DSN6SPRM	DSNTIPS	No
RESYNC	DSN6FAC	DSNTIPR	Yes
RETLWAIT	DSN6SPRM	DSNTIPI	Yes
RETVLCFK	DSN6SPRM	DSNTIP8	Yes
RGFCOLID	DSN6SPRM	DSNTIPZ	No
RGFDBNAM	DSN6SPRM	DSNTIPZ	No
RGFDEDPL	DSN6SPRM	DSNTIPZ	No
RGFDEFLT	DSN6SPRM	DSNTIPZ	No
RGFESCP	DSN6SPRM	DSNTIPZ	No
RGFFULLQ	DSN6SPRM	DSNTIPZ	No
RGFINSTL	DSN6SPRM	DSNTIPZ	No
RGFNMORT	DSN6SPRM	DSNTIPZ	No
RGFNMPRT	DSN6SPRM	DSNTIPZ	No
RLF	DSN6SYSP	DSNTIPO	No
RLFAUTH	DSN6SYSP	DSNTIPP	Yes
RLFERR	DSN6SYSP	DSNTIPO	Yes
RLFERRD	DSN6FAC	DSNTIPR	Yes
RLFTBL	DSN6SYSP	DSNTIPO	Yes
ROUTCDE	DSN6SYSP	DSNTIPO	No
RRULOCK	DSN6SPRM	DSNTIPI	Yes
SCCSID	DSNHDECP	DSNTIPF	—
SECQTY	DSN6ARVP	DSNTIPA	Yes
SEQCACH	DSN6SPRM	DSNTIPE	Yes
SEQPRES	DSN6SPRM	DSNTIPE	Yes
SITETYP	DSN6SPRM	DSNTIPO	No
SJMXPOOL	DSN6SPRM	DSNTIP8	Yes
SJTABLES	DSN6SPRM	—	Yes
SKIPUNCI	DSN6SPRM	DSNTIP8	Yes
SMF89	DSN6SYSP	—	Yes
SMFACCT	DSN6SYSP	DSNTIPN	No
SMFSTAT	DSN6SYSP	DSNTIPN	No
SMSDCFL	DSN6SPRM	—	Yes

Table 37. Directory of subsystem parameters and DSNHDECP values (continued)

	Parameter	Macro	Panel	Update Online
	SMSDCIX	DSN6SPRM	—	Yes
	SQLDELI	DSNHDECP	DSNTIPF	—
|	SRTPOOL	DSN6SPRM	DSNTIPC	Yes
	SSID	DSNHDECP	DSNTIPM	—
	STARJOIN	DSN6SPRM	DSNTIP8	Yes
	STATHIST	DSN6SPRM	DSNTIPO	Yes
	STATIME	DSN6SYSP	DSNTIPN	Yes
	STATROLL	DSN6SPRM	DSNTIPO	Yes
	STATSINT	DSN6SPRM	DSNTIPO	Yes
	STDSQL	DSNHDECP	DSNTIP4	—
	STORMXAB	DSN6SYSP	DSNTIPX	Yes
	STORPROC	DSN6SYSP	DSNTIPX	No
	STORTIME	DSN6SYSP	DSNTIPX	Yes
	SUPERRS	DSN6SPRM	DSNTIPM	Yes
#	SUPPRESS_TS_CONV_WARNING	DSN6SPRM	—	Yes
	SVOLARC	DSN6ARVP	DSNTIPA	Yes
	SYNCVAL	DSN6SYSP	DSNTIPN	Yes
|	SYSADM	DSN6SPRM	DSNTIPP	Yes
|	SYSADM2	DSN6SPRM	DSNTIPP	Yes
|	SYSOPR1	DSN6SPRM	DSNTIPP	Yes
|	SYSOPR2	DSN6SPRM	DSNTIPP	Yes
	TBSBPOOL	DSN6SYSP	DSNTIP1	Yes
|	TCPALVER	DSN6FAC	DSNTIP5	Yes
|	TCPKPALV	DSN6FAC	DSNTIP5	Yes
	TIME	DSNHDECP	DSNTIP4	—
	TIMELEN	DSNHDECP	DSNTIP4	—
	TRACSTR	DSN6SYSP	DSNTIPN	No
	TRACTBL	DSN6SYSP	DSNTIPN	No
	TRKRSITE	DSN6SPRM	DSNTIPO	No
	TSQTY	DSN6SYSP	DSNTIP7	Yes
	TSTAMP	DSN6ARVP	DSNTIPH	Yes
	TWOACTV	DSN6LOGP	DSNTIPH	No
	TWOARCH	DSN6LOGP	DSNTIPH	No
	TWOBSDS	DSN6LOGP	—	No
	UGCCSID	DSNHDECP	DSNTIPF	—
|	UIFCIDS	DSN6SYSP	DSNTIPN	Yes
	UMCCSID	DSNHDECP	DSNTIPF	—
#	UNION_COLNAME_7	DSN6SPRM	—	Yes
	UNIT	DSN6ARVP	DSNTIPA	Yes
	UNIT2	DSN6ARVP	DSNTIPA	Yes
	URCHKTH	DSN6SYSP	DSNTIPL	Yes
	URLGWTH	DSN6SYSP	DSNTIPL	Yes
	USCCSID	DSNHDECP	DSNTIPF	—
	UTIMOUT	DSN6SPRM	DSNTIPI	Yes
#	UTSORTAL	DSN6SPRM	—	Yes
	VOLTDEVT	DSN6SPRM	DSNTIPA2	Yes
	WLMENV	DSN6SYSP	DSNTIPX	Yes
|	XLKUPDLT	DSN6SPRM	DSNTIPI	Yes
#	XMLVALA	DSN6SYSP	—	Yes
#	XMLVALS	DSN6SYSP	—	Yes

Table 37. Directory of subsystem parameters and DSNHDECP values (continued)

Parameter	Macro	Panel	Update Online

How to use the DB2 library

Titles of books in the library begin with DB2 Universal Database for z/OS Version 8. However, references from one book in the library to another are shortened and do not include the product name, version, and release. Instead, they point directly to the section that holds the information. For a complete list of books in the library, and the sections in each book, see the bibliography at the back of this book.

The most rewarding task associated with a database management system is asking questions of it and getting answers, the task called *end use*. Other tasks are also necessary—defining the parameters of the system, putting the data in place, and so on. The tasks that are associated with DB2 are grouped into the following major categories (but supplemental information relating to all of the following tasks for new releases of DB2 can be found in *DB2 Release Planning Guide*.

Installation: If you are involved with DB2 only to install the system, *DB2 Installation Guide* might be all you need.

If you will be using data sharing capabilities you also need *DB2 Data Sharing: Planning and Administration*, which describes installation considerations for data sharing.

If you want to set up a DB2 subsystem to meet the requirements of the Common Criteria, you need *DB2 Common Criteria Guide*, which contains information that supersedes other information in the DB2 UDB for z/OS library regarding Common Criteria.

End use: End users issue SQL statements to retrieve data. They can also insert, update, or delete data, with SQL statements. They might need an introduction to SQL, detailed instructions for using SPUFI, and an alphabetized reference to the types of SQL statements. This information is found in *DB2 Application Programming and SQL Guide*, and *DB2 SQL Reference*.

End users can also issue SQL statements through the DB2 Query Management Facility (QMF) or some other program, and the library for that licensed program might provide all the instruction or reference material they need. For a list of the titles in the DB2 QMF library, see the bibliography at the end of this book.

Application programming: Some users access DB2 without knowing it, using programs that contain SQL statements. DB2 application programmers write those programs. Because they write SQL statements, they need the same resources that end users do.

Application programmers also need instructions on many other topics:

- How to transfer data between DB2 and a host program—written in Java, C, or COBOL, for example
- How to prepare to compile a program that embeds SQL statements
- How to process data from two systems simultaneously, say DB2 and IMS or DB2 and CICS
- How to write distributed applications across operating systemss
- How to write applications that use Open Database Connectivity (ODBC) to access DB2 servers

© Copyright IBM Corp. 1983, 2008

- How to write applications in the Java programming language to access DB2 servers

The material needed for writing a host program containing SQL is in *DB2 Application Programming and SQL Guide* and in *DB2 Application Programming Guide and Reference for Java*. The material needed for writing applications that use DB2 ODBC or ODBC to access DB2 servers is in *DB2 ODBC Guide and Reference*. For handling errors, see *DB2 Codes*.

If you will be working in a distributed environment, you will need *DB2 Reference for Remote DRDA Requesters and Servers*.

Information about writing applications across operating systems can be found in *IBM DB2 Universal Database SQL Reference for Cross-Platform Development*.

System and database administration: *Administration* covers almost everything else. *DB2 Administration Guide* divides those tasks among the following sections:

- Part 2 (Volume 1) of *DB2 Administration Guide* discusses the decisions that must be made when designing a database and tells how to implement the design by creating and altering DB2 objects, loading data, and adjusting to changes.
- Part 3 (Volume 1) of *DB2 Administration Guide* describes ways of controlling access to the DB2 system and to data within DB2, to audit aspects of DB2 usage, and to answer other security and auditing concerns.
- Part 4 (Volume 1) of *DB2 Administration Guide* describes the steps in normal day-to-day operation and discusses the steps one should take to prepare for recovery in the event of some failure.
- Part 5 (Volume 2) of *DB2 Administration Guide* explains how to monitor the performance of the DB2 system and its parts. It also lists things that can be done to make some parts run faster.

If you will be using the RACF access control module for DB2 authorization checking, you will need *DB2 RACF Access Control Module Guide*.

If you are involved with DB2 only to design the database, or plan operational procedures, you need *DB2 Administration Guide*. If you also want to carry out your own plans by creating DB2 objects, granting privileges, running utility jobs, and so on, you also need:

- *DB2 SQL Reference*, which describes the SQL statements you use to create, alter, and drop objects and grant and revoke privileges
- *DB2 Utility Guide and Reference*, which explains how to run utilities
- *DB2 Command Reference*, which explains how to run commands

If you will be using data sharing, you need *DB2 Data Sharing: Planning and Administration*, which describes how to plan for and implement data sharing.

Additional information about system and database administration can be found in *DB2 Messages* and *DB2 Codes*, which list messages and codes issued by DB2, with explanations and suggested responses.

Diagnosis: Diagnosticians detect and describe errors in the DB2 program. They might also recommend or apply a remedy. The documentation for this task is in *DB2 Diagnosis Guide and Reference*, *DB2 Messages*, and *DB2 Codes*.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

© Copyright IBM Corp. 1983, 2008

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Programming interface information

This book is intended to help you to use commands of DB2 Universal Database for z/OS (DB2 UDB for z/OS) and related subsystems. This book primarily documents General-use Programming Interface and Associated Guidance Information provided by DB2.

General-use programming interfaces allow the customer to write programs that obtain the services of DB2.

However, this book also documents Product-sensitive Programming Interface and Associated Guidance Information.

Product-sensitive programming interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of this IBM software product. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-sensitive programming interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed to run with new product releases or versions, or as a result of service.

Product-sensitive Programming Interface

Product-sensitive Programming Interface and Associated Guidance Information is identified where it occurs, by the following marking: Product-sensitive Programming Interface and Associated Guidance Information ...

End of Product-sensitive Programming Interface

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

AT	ibm.com
BookManager	IMS
C/370	iSeries
CICS	Language Environment
CICS Connection	MVS
DataPropagator	OpenEdition
DB2	OS/390
DB2 Universal Database	Parallel Sysplex
DFSMSdfp	PR/SM
DFSMSdss	QMF
DFSMShsm	RACF
Distributed Relational Database Architecture	Redbooks
DRDA	SAA
Enterprise Storage Server	SOM
ES/3090	System Object Model
eServer	System/390
FlashCopy	TotalStorage
IBM	VTAM
IBM Registry	WebSphere
	z/OS

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Glossary

The following terms and abbreviations are defined as they are used in the DB2 library.

A

abend. Abnormal end of task.

abend reason code. A 4-byte hexadecimal code that uniquely identifies a problem with DB2.

abnormal end of task (abend). Termination of a task, job, or subsystem because of an error condition that recovery facilities cannot resolve during execution.

access method services. The facility that is used to define and reproduce VSAM key-sequenced data sets.

access path. The path that is used to locate data that is specified in SQL statements. An access path can be indexed or sequential.

active log. The portion of the DB2 log to which log records are written as they are generated. The active log always contains the most recent log records, whereas the archive log holds those records that are older and no longer fit on the active log.

active member state. A state of a member of a data sharing group. The cross-system coupling facility identifies each active member with a group and associates the member with a particular task, address space, and z/OS system. A member that is not active has either a failed member state or a quiesced member state.

address space. A range of virtual storage pages that is identified by a number (ASID) and a collection of segment and page tables that map the virtual pages to real pages of the computer's memory.

address space connection. The result of connecting an allied address space to DB2. Each address space that contains a task that is connected to DB2 has exactly one address space connection, even though more than one task control block (TCB) can be present. See also *allied address space* and *task control block*.

| **address space identifier (ASID).** A unique
| system-assigned identifier for and address space.

administrative authority. A set of related privileges that DB2 defines. When you grant one of the administrative authorities to a person's ID, the person has all of the privileges that are associated with that administrative authority.

after trigger. A trigger that is defined with the trigger activation time AFTER.

agent. As used in DB2, the structure that associates all processes that are involved in a DB2 unit of work. An *allied agent* is generally synonymous with an *allied thread*. *System agents* are units of work that process tasks that are independent of the allied agent, such as prefetch processing, deferred writes, and service tasks.

aggregate function. An operation that derives its
result by using values from one or more rows. Contrast
with *scalar function*.

alias. An alternative name that can be used in SQL statements to refer to a table or view in the same or a remote DB2 subsystem.

allied address space. An area of storage that is external to DB2 and that is connected to DB2. An allied address space is capable of requesting DB2 services.

allied thread. A thread that originates at the local DB2 subsystem and that can access data at a remote DB2 subsystem.

allocated cursor. A cursor that is defined for stored procedure result sets by using the SQL ALLOCATE CURSOR statement.

already verified. An LU 6.2 security option that allows DB2 to provide the user's verified authorization ID when allocating a conversation. With this option, the user is not validated by the partner DB2 subsystem.

ambiguous cursor. A database cursor that is in a plan or package that contains either PREPARE or EXECUTE IMMEDIATE SQL statements, and for which the following statements are true: the cursor is not defined with the FOR READ ONLY clause or the FOR UPDATE OF clause; the cursor is not defined on a read-only result table; the cursor is not the target of a WHERE CURRENT clause on an SQL UPDATE or DELETE statement.

American National Standards Institute (ANSI). An organization consisting of producers, consumers, and general interest groups, that establishes the procedures by which accredited organizations create and maintain voluntary industry standards in the United States.

ANSI. American National Standards Institute.

APAR. Authorized program analysis report.

APAR fix corrective service. A temporary correction of an IBM software defect. The correction is temporary,

© Copyright IBM Corp. 1983, 2008

because it is usually replaced at a later date by a more permanent correction, such as a program temporary fix (PTF).

APF. Authorized program facility.

API. Application programming interface.

APPL. A VTAM network definition statement that is used to define DB2 to VTAM as an application program that uses SNA LU 6.2 protocols.

application. A program or set of programs that performs a task; for example, a payroll application.

application-directed connection. A connection that an application manages using the SQL CONNECT statement.

application plan. The control structure that is produced during the bind process. DB2 uses the application plan to process SQL statements that it encounters during statement execution.

application process. The unit to which resources and locks are allocated. An application process involves the execution of one or more programs.

application programming interface (API). A functional interface that is supplied by the operating system or by a separately orderable licensed program that allows an application program that is written in a high-level language to use specific data or functions of the operating system or licensed program.

application requester. The component on a remote system that generates DRDA requests for data on behalf of an application. An application requester accesses a DB2 database server using the DRDA application-directed protocol.

application server. The target of a request from a remote application. In the DB2 environment, the application server function is provided by the distributed data facility and is used to access DB2 data from remote applications.

archive log. The portion of the DB2 log that contains log records that have been copied from the active log.

ASCII. An encoding scheme that is used to represent strings in many environments, typically on PCs and workstations. Contrast with *EBCDIC* and *Unicode*.

ASID. Address space identifier.

attachment facility. An interface between DB2 and TSO, IMS, CICS, or batch address spaces. An attachment facility allows application programs to access DB2.

attribute. A characteristic of an entity. For example, in database design, the phone number of an employee is one of that employee's attributes.

authorization ID. A string that can be verified for connection to DB2 and to which a set of privileges is allowed. It can represent an individual, an organizational group, or a function, but DB2 does not determine this representation.

authorized program analysis report (APAR). A report of a problem that is caused by a suspected defect in a current release of an IBM supplied program.

authorized program facility (APF). A facility that permits the identification of programs that are authorized to use restricted functions.

automatic query rewrite. A process that examines an SQL statement that refers to one or more base tables, and, if appropriate, rewrites the query so that it performs better. This process can also determine whether to rewrite a query so that it refers to one or more materialized query tables that are derived from the source tables.

auxiliary index. An index on an auxiliary table in which each index entry refers to a LOB.

auxiliary table. A table that stores columns outside the table in which they are defined. Contrast with *base table*.

B

backout. The process of undoing uncommitted changes that an application process made. This might be necessary in the event of a failure on the part of an application process, or as a result of a deadlock situation.

backward log recovery. The fourth and final phase of restart processing during which DB2 scans the log in a backward direction to apply UNDO log records for all aborted changes.

base table. (1) A table that is created by the SQL CREATE TABLE statement and that holds persistent data. Contrast with *result table* and *temporary table*.

(2) A table containing a LOB column definition. The actual LOB column data is not stored with the base table. The base table contains a row identifier for each row and an indicator column for each of its LOB columns. Contrast with *auxiliary table*.

base table space. A table space that contains base tables.

basic predicate. A predicate that compares two values.

basic sequential access method (BSAM). An access method for storing or retrieving data blocks in a continuous sequence, using either a sequential-access or a direct-access device.

batch message processing program. In IMS, an application program that can perform batch-type processing online and can access the IMS input and output message queues.

before trigger. A trigger that is defined with the trigger activation time BEFORE.

binary integer. A basic data type that can be further classified as small integer or large integer.

binary large object (BLOB). A sequence of bytes in which the size of the value ranges from 0 bytes to 2 GB–1. Such a string has a CCSID value of 65535.

binary string. A sequence of bytes that is not associated with a CCSID. For example, the BLOB data type is a binary string.

bind. The process by which the output from the SQL precompiler is converted to a usable control structure, often called an access plan, application plan, or package. During this process, access paths to the data are selected and some authorization checking is performed. The types of bind are:

automatic bind. (More correctly, *automatic rebind*) A process by which SQL statements are bound automatically (without a user issuing a BIND command) when an application process begins execution and the bound application plan or package it requires is not valid.

dynamic bind. A process by which SQL statements are bound as they are entered.

incremental bind. A process by which SQL statements are bound during the execution of an application process.

static bind. A process by which SQL statements are bound after they have been precompiled. All static SQL statements are prepared for execution at the same time.

bit data. Data that is character type CHAR or VARCHAR and has a CCSID value of 65535.

BLOB. Binary large object.

block fetch. A capability in which DB2 can retrieve, or fetch, a large set of rows together. Using block fetch can significantly reduce the number of messages that are being sent across the network. Block fetch applies only to cursors that do not update data.

BMP. Batch Message Processing (IMS). See *batch message processing program*.

bootstrap data set (BSDS). A VSAM data set that contains name and status information for DB2, as well as RBA range specifications, for all active and archive log data sets. It also contains passwords for the DB2 directory and catalog, and lists of conditional restart and checkpoint records.

BSAM. Basic sequential access method.

BSDS. Bootstrap data set.

buffer pool. Main storage that is reserved to satisfy the buffering requirements for one or more table spaces or indexes.

built-in data type. A data type that IBM supplies. Among the built-in data types for DB2 UDB for z/OS are string, numeric, ROWID, and datetime. Contrast with *distinct type*.

built-in function. A function that DB2 supplies. Contrast with *user-defined function*.

business dimension. A category of data, such as products or time periods, that an organization might want to analyze.

C

cache structure. A coupling facility structure that stores data that can be available to all members of a Sysplex. A DB2 data sharing group uses cache structures as group buffer pools.

CAF. Call attachment facility.

call attachment facility (CAF). A DB2 attachment facility for application programs that run in TSO or z/OS batch. The CAF is an alternative to the DSN command processor and provides greater control over the execution environment.

call-level interface (CLI). A callable application programming interface (API) for database access, which is an alternative to using embedded SQL. In contrast to embedded SQL, DB2 ODBC (which is based on the CLI architecture) does not require the user to precompile or bind applications, but instead provides a standard set of functions to process SQL statements and related services at run time.

cascade delete. The way in which DB2 enforces referential constraints when it deletes all descendent rows of a deleted parent row.

CASE expression. An expression that is selected based on the evaluation of one or more conditions.

cast function. A function that is used to convert instances of a (source) data type into instances of a different (target) data type. In general, a cast function has the name of the target data type. It has one single argument whose type is the source data type; its return type is the target data type.

castout. The DB2 process of writing changed pages from a group buffer pool to disk.

castout owner. The DB2 member that is responsible for casting out a particular page set or partition.

catalog. In DB2, a collection of tables that contains descriptions of objects such as tables, views, and indexes.

catalog table. Any table in the DB2 catalog.

CCSID. Coded character set identifier.

CDB. Communications database.

CDRA. Character Data Representation Architecture.

CEC. Central electronic complex. See *central processor complex*.

central electronic complex (CEC). See *central processor complex*.

central processor (CP). The part of the computer that contains the sequencing and processing facilities for instruction execution, initial program load, and other machine operations.

central processor complex (CPC). A physical collection of hardware (such as an ES/3090™) that consists of main storage, one or more central processors, timers, and channels.

| **CFRM.** Coupling facility resource management.

CFRM policy. A declaration by a z/OS administrator regarding the allocation rules for a coupling facility structure.

character conversion. The process of changing characters from one encoding scheme to another.

Character Data Representation Architecture (CDRA). An architecture that is used to achieve consistent representation, processing, and interchange of string data.

character large object (CLOB). A sequence of bytes representing single-byte characters or a mixture of single- and double-byte characters where the size of the value can be up to 2 GB–1. In general, character large object values are used whenever a character string might exceed the limits of the VARCHAR type.

character set. A defined set of characters.

character string. A sequence of bytes that represent bit data, single-byte characters, or a mixture of single-byte and multibyte characters.

check constraint. A user-defined constraint that specifies the values that specific columns of a base table can contain.

check integrity. The condition that exists when each row in a table conforms to the check constraints that are defined on that table. Maintaining check integrity requires DB2 to enforce check constraints on operations that add or change data.

check pending. A state of a table space or partition that prevents its use by some utilities and by some SQL statements because of rows that violate referential constraints, check constraints, or both.

checkpoint. A point at which DB2 records internal status information on the DB2 log; the recovery process uses this information if DB2 abnormally terminates.

| **child lock.** For explicit hierarchical locking, a lock that
| is held on either a table, page, row, or a large object
| (LOB). Each child lock has a parent lock. See also *parent*
| *lock*.

CI. Control interval.

| **CICS.** Represents (in this publication): CICS
| Transaction Server for z/OS: Customer Information
| Control System Transaction Server for z/OS.

CICS attachment facility. A DB2 subcomponent that uses the z/OS subsystem interface (SSI) and cross-storage linkage to process requests from CICS to DB2 and to coordinate resource commitment.

CIDF. Control interval definition field.

claim. A notification to DB2 that an object is being accessed. Claims prevent drains from occurring until the claim is released, which usually occurs at a commit point. Contrast with *drain*.

claim class. A specific type of object access that can be one of the following isolation levels:
Cursor stability (CS)
Repeatable read (RR)
Write

claim count. A count of the number of agents that are accessing an object.

class of service. A VTAM term for a list of routes through a network, arranged in an order of preference for their use.

class word. A single word that indicates the nature of a data attribute. For example, the class word PROJ indicates that the attribute identifies a project.

clause. In SQL, a distinct part of a statement, such as a SELECT clause or a WHERE clause.

CLI. Call- level interface.

client. See *requester*.

CLIST. Command list. A language for performing TSO tasks.

CLOB. Character large object.

closed application. An application that requires exclusive use of certain statements on certain DB2

objects, so that the objects are managed solely through the application's external interface.

CLPA. Create link pack area.

clustering index. An index that determines how rows are physically ordered (*clustered*) in a table space. If a clustering index on a partitioned table is not a partitioning index, the rows are ordered in cluster sequence within each data partition instead of spanning partitions. Prior to Version 8 of DB2 UDB for z/OS, the partitioning index was required to be the clustering index.

coded character set. A set of unambiguous rules that establish a character set and the one-to-one relationships between the characters of the set and their coded representations.

coded character set identifier (CCSID). A 16-bit number that uniquely identifies a coded representation of graphic characters. It designates an encoding scheme identifier and one or more pairs consisting of a character set identifier and an associated code page identifier.

code page. (1) A set of assignments of characters to code points. In EBCDIC, for example, the character 'A' is assigned code point X'C1' (2) , and character 'B' is assigned code point X'C2'. Within a code page, each code point has only one specific meaning.

code point. In CDRA, a unique bit pattern that represents a character in a code page.

code unit. The fundamental binary width in a
computer architecture that is used for representing
character data, such as 7 bits, 8 bits, 16 bits, or 32 bits.
Depending on the character encoding form that is used,
each code point in a coded character set can be
represented internally by one or more code units.

coexistence. During migration, the period of time in which two releases exist in the same data sharing group.

cold start. A process by which DB2 restarts without processing any log records. Contrast with *warm start*.

collection. A group of packages that have the same qualifier.

column. The vertical component of a table. A column has a name and a particular data type (for example, character, decimal, or integer).

column function. See *aggregate function*.

"come from" checking. An LU 6.2 security option that defines a list of authorization IDs that are allowed to connect to DB2 from a partner LU.

command. A DB2 operator command or a DSN subcommand. A command is distinct from an SQL statement.

command prefix. A one- to eight-character command identifier. The command prefix distinguishes the command as belonging to an application or subsystem rather than to MVS.

command recognition character (CRC). A character that permits a z/OS console operator or an IMS subsystem user to route DB2 commands to specific DB2 subsystems.

command scope. The scope of command operation in a data sharing group. If a command has *member scope*, the command displays information only from the one member or affects only non-shared resources that are owned locally by that member. If a command has *group scope*, the command displays information from all members, affects non-shared resources that are owned locally by all members, displays information on sharable resources, or affects sharable resources.

commit. The operation that ends a unit of work by releasing locks so that the database changes that are made by that unit of work can be perceived by other processes.

commit point. A point in time when data is considered consistent.

committed phase. The second phase of the multisite update process that requests all participants to commit the effects of the logical unit of work.

common service area (CSA). In z/OS, a part of the common area that contains data areas that are addressable by all address spaces.

communications database (CDB). A set of tables in the DB2 catalog that are used to establish conversations with remote database management systems.

comparison operator. A token (such as =, >, or <) that is used to specify a relationship between two values.

composite key. An ordered set of key columns of the same table.

compression dictionary. The dictionary that controls the process of compression and decompression. This dictionary is created from the data in the table space or table space partition.

concurrency. The shared use of resources by more than one application process at the same time.

conditional restart. A DB2 restart that is directed by a user-defined conditional restart control record (CRCR).

connection. In SNA, the existence of a communication path between two partner LUs that allows information

to be exchanged (for example, two DB2 subsystems that are connected and communicating by way of a conversation).

connection context. In SQLJ, a Java™ object that represents a connection to a data source.

connection declaration clause. In SQLJ, a statement that declares a connection to a data source.

connection handle. The data object containing information that is associated with a connection that DB2 ODBC manages. This includes general status information, transaction status, and diagnostic information.

connection ID. An identifier that is supplied by the attachment facility and that is associated with a specific address space connection.

consistency token. A timestamp that is used to generate the version identifier for an application. See also *version*.

constant. A language element that specifies an unchanging value. Constants are classified as string constants or numeric constants. Contrast with *variable*.

constraint. A rule that limits the values that can be inserted, deleted, or updated in a table. See *referential constraint*, *check constraint*, and *unique constraint*.

context. The application's logical connection to the data source and associated internal DB2 ODBC connection information that allows the application to direct its operations to a data source. A DB2 ODBC context represents a DB2 thread.

contracting conversion. A process that occurs when the length of a converted string is smaller than that of the source string. For example, this process occurs when an EBCDIC mixed-data string that contains DBCS characters is converted to ASCII mixed data; the converted string is shorter because of the removal of the shift codes.

control interval (CI). A fixed-length area or disk in which VSAM stores records and creates distributed free space. Also, in a key-sequenced data set or file, the set of records that an entry in the sequence-set index record points to. The control interval is the unit of information that VSAM transmits to or from disk. A control interval always includes an integral number of physical records.

control interval definition field (CIDF). In VSAM, a field that is located in the 4 bytes at the end of each control interval; it describes the free space, if any, in the control interval.

conversation. Communication, which is based on LU 6.2 or Advanced Program-to-Program Communication (APPC), between an application and a remote transaction program over an SNA logical unit-to-logical unit (LU-LU) session that allows communication while processing a transaction.

coordinator. The system component that coordinates the commit or rollback of a unit of work that includes work that is done on one or more other systems.

copy pool. A named set of SMS storage groups that contains data that is to be copied collectively. A copy pool is an SMS construct that lets you define which storage groups are to be copied by using FlashCopy functions. HSM determines which volumes belong to a copy pool.

copy target. A named set of SMS storage groups that are to be used as containers for copy pool volume copies. A copy target is an SMS construct that lets you define which storage groups are to be used as containers for volumes that are copied by using FlashCopy functions.

copy version. A point-in-time FlashCopy copy that is managed by HSM. Each copy pool has a version parameter that specifies how many copy versions are maintained on disk.

correlated columns. A relationship between the value of one column and the value of another column.

correlated subquery. A subquery (part of a WHERE or HAVING clause) that is applied to a row or group of rows of a table or view that is named in an outer subselect statement.

correlation ID. An identifier that is associated with a specific thread. In TSO, it is either an authorization ID or the job name.

correlation name. An identifier that designates a table, a view, or individual rows of a table or view within a single SQL statement. It can be defined in any FROM clause or in the first clause of an UPDATE or DELETE statement.

cost category. A category into which DB2 places cost estimates for SQL statements at the time the statement is bound. A cost estimate can be placed in either of the following cost categories:

- A: Indicates that DB2 had enough information to make a cost estimate without using default values.
- B: Indicates that some condition exists for which DB2 was forced to use default values for its estimate.

The cost category is externalized in the COST_CATEGORY column of the DSN_STATEMNT_TABLE when a statement is explained.

coupling facility. A special PR/SM™ LPAR logical partition that runs the coupling facility control program and provides high-speed caching, list processing, and locking functions in a Parallel Sysplex®.

coupling facility resource management. A component of z/OS that provides the services to manage coupling facility resources in a Parallel Sysplex. This management includes the enforcement of CFRM policies to ensure that the coupling facility and structure requirements are satisfied.

CP. Central processor.

CPC. Central processor complex.

C++ member. A data object or function in a structure, union, or class.

C++ member function. An operator or function that is declared as a member of a class. A member function has access to the private and protected data members and to the member functions of objects in its class. Member functions are also called methods.

C++ object. (1) A region of storage. An object is created when a variable is defined or a new function is invoked. (2) An instance of a class.

CRC. Command recognition character.

CRCR. Conditional restart control record. See also *conditional restart*.

create link pack area (CLPA). An option that is used during IPL to initialize the link pack pageable area.

created temporary table. A table that holds temporary data and is defined with the SQL statement CREATE GLOBAL TEMPORARY TABLE. Information about created temporary tables is stored in the DB2 catalog, so this kind of table is persistent and can be shared across application processes. Contrast with *declared temporary table*. See also *temporary table*.

cross-memory linkage. A method for invoking a program in a different address space. The invocation is synchronous with respect to the caller.

cross-system coupling facility (XCF). A component of z/OS that provides functions to support cooperation between authorized programs that run within a Sysplex.

cross-system extended services (XES). A set of z/OS services that allow multiple instances of an application or subsystem, running on different systems in a Sysplex environment, to implement high-performance, high-availability data sharing by using a coupling facility.

CS. Cursor stability.

CSA. Common service area.

CT. Cursor table.

current data. Data within a host structure that is current with (identical to) the data within the base table.

current SQL ID. An ID that, at a single point in time, holds the privileges that are exercised when certain dynamic SQL statements run. The current SQL ID can be a primary authorization ID or a secondary authorization ID.

current status rebuild. The second phase of restart processing during which the status of the subsystem is reconstructed from information on the log.

cursor. A named control structure that an application program uses to point to a single row or multiple rows within some ordered set of rows of a result table. A cursor can be used to retrieve, update, or delete rows from a result table.

cursor sensitivity. The degree to which database updates are visible to the subsequent FETCH statements in a cursor. A cursor can be sensitive to changes that are made with positioned update and delete statements specifying the name of that cursor. A cursor can also be sensitive to changes that are made with searched update or delete statements, or with cursors other than this cursor. These changes can be made by this application process or by another application process.

cursor stability (CS). The isolation level that provides maximum concurrency without the ability to read uncommitted data. With cursor stability, a unit of work holds locks only on its uncommitted changes and on the current row of each of its cursors.

cursor table (CT). The copy of the skeleton cursor table that is used by an executing application process.

cycle. A set of tables that can be ordered so that each table is a descendent of the one before it, and the first table is a descendent of the last table. A self-referencing table is a cycle with a single member.

D

DAD. See *Document access definition.*

disk. A direct-access storage device that records data magnetically.

database. A collection of tables, or a collection of table spaces and index spaces.

database access thread. A thread that accesses data at the local subsystem on behalf of a remote subsystem.

database administrator (DBA). An individual who is responsible for designing, developing, operating, safeguarding, maintaining, and using a database.

database alias. The name of the target server if different from the location name. The database alias name is used to provide the name of the database server as it is known to the network. When a database alias name is defined, the location name is used by the application to reference the server, but the database alias name is used to identify the database server to be accessed. Any fully qualified object names within any SQL statements are not modified and are sent unchanged to the database server.

database descriptor (DBD). An internal representation of a DB2 database definition, which reflects the data definition that is in the DB2 catalog. The objects that are defined in a database descriptor are table spaces, tables, indexes, index spaces, relationships, check constraints, and triggers. A DBD also contains information about accessing tables in the database.

database exception status. An indication that something is wrong with a database. All members of a data sharing group must know and share the exception status of databases.

database identifier (DBID). An internal identifier of the database.

database management system (DBMS). A software system that controls the creation, organization, and modification of a database and the access to the data that is stored within it.

database request module (DBRM). A data set member that is created by the DB2 precompiler and that contains information about SQL statements. DBRMs are used in the bind process.

database server. The target of a request from a local application or an intermediate database server. In the DB2 environment, the database server function is provided by the distributed data facility to access DB2 data from local applications, or from a remote database server that acts as an intermediate database server.

data currency. The state in which data that is retrieved into a host variable in your program is a copy of data in the base table.

data definition name (ddname). The name of a data definition (DD) statement that corresponds to a data control block containing the same name.

data dictionary. A repository of information about an organization's application programs, databases, logical data models, users, and authorizations. A data dictionary can be manual or automated.

data-driven business rules. Constraints on particular data values that exist as a result of requirements of the business.

Data Language/I (DL/I). The IMS data manipulation language; a common high-level interface between a user application and IMS.

data mart. A small data warehouse that applies to a single department or team. See also *data warehouse*.

data mining. The process of collecting critical business information from a data warehouse, correlating it, and uncovering associations, patterns, and trends.

data partition. A VSAM data set that is contained within a partitioned table space.

data-partitioned secondary index (DPSI). A secondary index that is partitioned. The index is partitioned according to the underlying data.

data sharing. The ability of two or more DB2 subsystems to directly access and change a single set of data.

data sharing group. A collection of one or more DB2 subsystems that directly access and change the same data while maintaining data integrity.

data sharing member. A DB2 subsystem that is assigned by XCF services to a data sharing group.

data source. A local or remote relational or non-relational data manager that is capable of supporting data access via an ODBC driver that supports the ODBC APIs. In the case of DB2 UDB for z/OS, the data sources are always relational database managers.

data space. In releases prior to DB2 UDB for z/OS, Version 8, a range of up to 2 GB of contiguous virtual storage addresses that a program can directly manipulate. Unlike an address space, a data space can hold only data; it does not contain common areas, system data, or programs.

data type. An attribute of columns, literals, host variables, special registers, and the results of functions and expressions.

data warehouse. A system that provides critical business information to an organization. The data warehouse system cleanses the data for accuracy and currency, and then presents the data to decision makers so that they can interpret and use it effectively and efficiently.

date. A three-part value that designates a day, month, and year.

date duration. A decimal integer that represents a number of years, months, and days.

datetime value. A value of the data type DATE, TIME, or TIMESTAMP.

DBA. Database administrator.

DBCLOB. Double-byte character large object.

DBCS. Double-byte character set.

DBD. Database descriptor.

DBID. Database identifier.

DBMS. Database management system.

DBRM. Database request module.

DB2 catalog. Tables that are maintained by DB2 and contain descriptions of DB2 objects, such as tables, views, and indexes.

DB2 command. An instruction to the DB2 subsystem that a user enters to start or stop DB2, to display information on current users, to start or stop databases, to display information on the status of databases, and so on.

DB2 for VSE & VM. The IBM DB2 relational database management system for the VSE and VM operating systems.

DB2I. DB2 Interactive.

DB2 Interactive (DB2I). The DB2 facility that provides for the execution of SQL statements, DB2 (operator) commands, programmer commands, and utility invocation.

DB2I Kanji Feature. The tape that contains the panels and jobs that allow a site to display DB2I panels in Kanji.

DB2 PM. DB2 Performance Monitor.

DB2 thread. The DB2 structure that describes an application's connection, traces its progress, processes resource functions, and delimits its accessibility to DB2 resources and services.

DCLGEN. Declarations generator.

DDF. Distributed data facility.

ddname. Data definition name.

deadlock. Unresolvable contention for the use of a resource, such as a table or an index.

declarations generator (DCLGEN). A subcomponent of DB2 that generates SQL table declarations and COBOL, C, or PL/I data structure declarations that conform to the table. The declarations are generated from DB2 system catalog information. DCLGEN is also a DSN subcommand.

declared temporary table. A table that holds temporary data and is defined with the SQL statement DECLARE GLOBAL TEMPORARY TABLE. Information about declared temporary tables is not stored in the DB2 catalog, so this kind of table is not persistent and can be used only by the application process that issued the DECLARE statement. Contrast with *created temporary table*. See also *temporary table*.

default value. A predetermined value, attribute, or option that is assumed when no other is explicitly specified.

deferred embedded SQL. SQL statements that are neither fully static nor fully dynamic. Like static statements, they are embedded within an application, but like dynamic statements, they are prepared during the execution of the application.

deferred write. The process of asynchronously writing changed data pages to disk.

degree of parallelism. The number of concurrently executed operations that are initiated to process a query.

delete-connected. A table that is a dependent of table P or a dependent of a table to which delete operations from table P cascade.

delete hole. The location on which a cursor is positioned when a row in a result table is refetched and the row no longer exists on the base table, because another cursor deleted the row between the time the cursor first included the row in the result table and the time the cursor tried to refetch it.

delete rule. The rule that tells DB2 what to do to a dependent row when a parent row is deleted. For each relationship, the rule might be CASCADE, RESTRICT, SET NULL, or NO ACTION.

delete trigger. A trigger that is defined with the triggering SQL operation DELETE.

delimited identifier. A sequence of characters that are enclosed within double quotation marks ("). The sequence must consist of a letter followed by zero or more characters, each of which is a letter, digit, or the underscore character (_).

delimiter token. A string constant, a delimited identifier, an operator symbol, or any of the special characters that are shown in DB2 syntax diagrams.

denormalization. A key step in the task of building a physical relational database design. Denormalization is the intentional duplication of columns in multiple tables, and the consequence is increased data redundancy. Denormalization is sometimes necessary to minimize performance problems. Contrast with *normalization*.

dependent. An object (row, table, or table space) that has at least one parent. The object is also said to be a dependent (row, table, or table space) of its parent. See also *parent row*, *parent table*, *parent table space*.

dependent row. A row that contains a foreign key that matches the value of a primary key in the parent row.

dependent table. A table that is a dependent in at least one referential constraint.

DES-based authenticator. An authenticator that is generated using the DES algorithm.

descendent. An object that is a dependent of an object or is the dependent of a descendent of an object.

descendent row. A row that is dependent on another row, or a row that is a descendent of a dependent row.

descendent table. A table that is a dependent of another table, or a table that is a descendent of a dependent table.

deterministic function. A user-defined function whose result is dependent on the values of the input arguments. That is, successive invocations with the same input values produce the same answer. Sometimes referred to as a *not-variant* function. Contrast this with an *nondeterministic function* (sometimes called a *variant function*), which might not always produce the same result for the same inputs.

DFP. Data Facility Product (in z/OS).

DFSMS. Data Facility Storage Management Subsystem (in z/OS). Also called *Storage Management Subsystem (SMS)*.

| **DFSMSdss™.** The data set services (dss) component of
| DFSMS (in z/OS).

| **DFSMShsm™.** The hierarchical storage manager (hsm)
| component of DFSMS (in z/OS).

dimension. A data category such as time, products, or markets. The elements of a dimension are referred to as members. Dimensions offer a very concise, intuitive way of organizing and selecting data for retrieval, exploration, and analysis. See also *dimension table*.

dimension table. The representation of a dimension in a star schema. Each row in a dimension table represents all of the attributes for a particular member of the dimension. See also *dimension*, *star schema*, and *star join*.

directory. The DB2 system database that contains internal objects such as database descriptors and skeleton cursor tables.

distinct predicate. In SQL, a predicate that ensures
that two row values are not equal, and that both row
values are not null.

distinct type. A user-defined data type that is internally represented as an existing type (its source type), but is considered to be a separate and incompatible type for semantic purposes.

distributed data. Data that resides on a DBMS other than the local system.

distributed data facility (DDF). A set of DB2 components through which DB2 communicates with another relational database management system.

Distributed Relational Database Architecture™ (DRDA). A connection protocol for distributed relational database processing that is used by IBM's relational database products. DRDA includes protocols for communication between an application and a remote relational database management system, and for communication between relational database management systems. See also *DRDA access*.

DL/I. Data Language/I.

DNS. Domain name server.

| **document access definition (DAD).** Used to define
| the indexing scheme for an XML column or the
| mapping scheme of an XML collection. It can be used
| to enable an XML Extender column of an XML
| collection, which is XML formatted.

domain. The set of valid values for an attribute.

domain name. The name by which TCP/IP applications refer to a TCP/IP host within a TCP/IP network.

domain name server (DNS). A special TCP/IP network server that manages a distributed directory that is used to map TCP/IP host names to IP addresses.

double-byte character large object (DBCLOB). A sequence of bytes representing double-byte characters where the size of the values can be up to 2 GB. In general, DBCLOB values are used whenever a double-byte character string might exceed the limits of the VARGRAPHIC type.

double-byte character set (DBCS). A set of characters, which are used by national languages such as Japanese and Chinese, that have more symbols than can be represented by a single byte. Each character is 2 bytes in length. Contrast with *single-byte character set* and *multibyte character set*.

double-precision floating point number. A 64-bit approximate representation of a real number.

downstream. The set of nodes in the syncpoint tree that is connected to the local DBMS as a participant in the execution of a two-phase commit.

| **DPSI.** Data-partitioned secondary index.

drain. The act of acquiring a locked resource by quiescing access to that object.

drain lock. A lock on a claim class that prevents a claim from occurring.

DRDA. Distributed Relational Database Architecture.

DRDA access. An open method of accessing distributed data that you can use to can connect to another database server to execute packages that were previously bound at the server location. You use the SQL CONNECT statement or an SQL statement with a three-part name to identify the server. Contrast with *private protocol access*.

DSN. (1) The default DB2 subsystem name. (2) The name of the TSO command processor of DB2. (3) The first three characters of DB2 module and macro names.

duration. A number that represents an interval of time. See also *date duration*, *labeled duration*, and *time duration*.

dynamic cursor. A named control structure that an application program uses to change the size of the result table and the order of its rows after the cursor is opened. Contrast with *static cursor.*

dynamic dump. A dump that is issued during the execution of a program, usually under the control of that program.

dynamic SQL. SQL statements that are prepared and executed within an application program while the program is executing. In dynamic SQL, the SQL source is contained in host language variables rather than being coded into the application program. The SQL statement can change several times during the application program's execution.

dynamic statement cache pool. A cache, located above the 2-GB storage line, that holds dynamic statements.

E

EA-enabled table space. A table space or index space that is enabled for extended addressability and that contains individual partitions (or pieces, for LOB table spaces) that are greater than 4 GB.

EB. See *exabyte*.

EBCDIC. Extended binary coded decimal interchange code. An encoding scheme that is used to represent character data in the z/OS, VM, VSE, and iSeries™ environments. Contrast with *ASCII* and *Unicode*.

e-business. The transformation of key business processes through the use of Internet technologies.

EDM pool. A pool of main storage that is used for database descriptors, application plans, authorization cache, application packages.

EID. Event identifier.

embedded SQL. SQL statements that are coded within an application program. See *static SQL*.

enclave. In Language Environment , an independent collection of routines, one of which is designated as the main routine. An enclave is similar to a program or run unit.

encoding scheme. A set of rules to represent character data (ASCII, EBCDIC, or Unicode).

entity. A significant object of interest to an organization.

enumerated list. A set of DB2 objects that are defined with a LISTDEF utility control statement in which pattern-matching characters (*, %, _ or ?) are not used.

environment. A collection of names of logical and physical resources that are used to support the performance of a function.

environment handle. In DB2 ODBC, the data object that contains global information regarding the state of the application. An environment handle must be allocated before a connection handle can be allocated. Only one environment handle can be allocated per application.

EOM. End of memory.

EOT. End of task.

equijoin. A join operation in which the join-condition has the form *expression = expression*.

error page range. A range of pages that are considered to be physically damaged. DB2 does not allow users to access any pages that fall within this range.

escape character. The symbol that is used to enclose an SQL delimited identifier. The escape character is the double quotation mark ("), except in COBOL applications, where the user assigns the symbol, which is either a double quotation mark or an apostrophe (').

ESDS. Entry sequenced data set.

ESMT. External subsystem module table (in IMS).

EUR. IBM European Standards.

exabyte. For processor, real and virtual storage capacities and channel volume:
1 152 921 504 606 846 976 bytes or 2^{60}.

exception table. A table that holds rows that violate referential constraints or check constraints that the CHECK DATA utility finds.

exclusive lock. A lock that prevents concurrently executing application processes from reading or changing data. Contrast with *share lock*.

executable statement. An SQL statement that can be embedded in an application program, dynamically prepared and executed, or issued interactively.

execution context. In SQLJ, a Java object that can be used to control the execution of SQL statements.

exit routine. A user-written (or IBM-provided default) program that receives control from DB2 to perform specific functions. Exit routines run as extensions of DB2.

expanding conversion. A process that occurs when the length of a converted string is greater than that of the source string. For example, this process occurs when an ASCII mixed-data string that contains DBCS characters is converted to an EBCDIC mixed-data string; the converted string is longer because of the addition of shift codes.

explicit hierarchical locking. Locking that is used to make the parent-child relationship between resources known to IRLM. This kind of locking avoids global locking overhead when no inter-DB2 interest exists on a resource.

exposed name. A correlation name or a table or view name for which a correlation name is not specified. Names that are specified in a FROM clause are exposed or non-exposed.

expression. An operand or a collection of operators and operands that yields a single value.

extended recovery facility (XRF). A facility that minimizes the effect of failures in z/OS, VTAM , the host processor, or high-availability applications during sessions between high-availability applications and designated terminals. This facility provides an alternative subsystem to take over sessions from the failing subsystem.

Extensible Markup Language (XML). A standard metalanguage for defining markup languages that is a subset of Standardized General Markup Language (SGML). The less complex nature of XML makes it easier to write applications that handle document types, to author and manage structured information, and to transmit and share structured information across diverse computing environments.

external function. A function for which the body is written in a programming language that takes scalar argument values and produces a scalar result for each invocation. Contrast with *sourced function*, *built-in function*, and *SQL function*.

external procedure. A user-written application program that can be invoked with the SQL CALL statement, which is written in a programming language. Contrast with *SQL procedure*.

external routine. A user-defined function or stored procedure that is based on code that is written in an external programming language.

external subsystem module table (ESMT). In IMS, the table that specifies which attachment modules must be loaded.

F

failed member state. A state of a member of a data sharing group. When a member fails, the XCF permanently records the failed member state. This state usually means that the member's task, address space, or z/OS system terminated before the state changed from active to quiesced.

fallback. The process of returning to a previous release of DB2 after attempting or completing migration to a current release.

false global lock contention. A contention indication from the coupling facility when multiple lock names are hashed to the same indicator and when no real contention exists.

fan set. A direct physical access path to data, which is provided by an index, hash, or link; a fan set is the means by which the data manager supports the ordering of data.

federated database. The combination of a DB2 Universal Database server (in Linux, UNIX®, and Windows® environments) and multiple data sources to which the server sends queries. In a federated database system, a client application can use a single SQL statement to join data that is distributed across multiple database management systems and can view the data as if it were local.

fetch orientation. The specification of the desired placement of the cursor as part of a FETCH statement (for example, BEFORE, AFTER, NEXT, PRIOR, CURRENT, FIRST, LAST, ABSOLUTE, and RELATIVE).

field procedure. A user-written exit routine that is designed to receive a single value and transform (encode or decode) it in any way the user can specify.

filter factor. A number between zero and one that estimates the proportion of rows in a table for which a predicate is true.

fixed-length string. A character or graphic string whose length is specified and cannot be changed. Contrast with *varying-length string*.

FlashCopy. A function on the IBM Enterprise Storage Server® that can create a point-in-time copy of data while an application is running.

foreign key. A column or set of columns in a dependent table of a constraint relationship. The key must have the same number of columns, with the same descriptions, as the primary key of the parent table.

Each foreign key value must either match a parent key value in the related parent table or be null.

forest. An ordered set of subtrees of XML nodes.

forget. In a two-phase commit operation, (1) the vote that is sent to the prepare phase when the participant has not modified any data. The forget vote allows a participant to release locks and forget about the logical unit of work. This is also referred to as the read-only vote. (2) The response to the *committed* request in the second phase of the operation.

forward log recovery. The third phase of restart processing during which DB2 processes the log in a forward direction to apply all REDO log records.

free space. The total amount of unused space in a page; that is, the space that is not used to store records or control information is free space.

full outer join. The result of a join operation that includes the matched rows of both tables that are being joined and preserves the unmatched rows of both tables. See also *join*.

fullselect. A subselect, a values-clause, or a number of both that are combined by set operators. *Fullselect* specifies a result table. If UNION is not used, the result of the fullselect is the result of the specified subselect.

fully escaped mapping. A mapping from an SQL identifier to an XML name when the SQL identifier is a column name.

function. A mapping, which is embodied as a program (the function body) that is invocable by means of zero or more input values (arguments) to a single value (the result). See also *aggregate function* and *scalar function*.

Functions can be user-defined, built-in, or generated by DB2. (See also *built-in function*, *cast function*, *external function*, *sourced function*, *SQL function*, and *user-defined function*.)

function definer. The authorization ID of the owner of the schema of the function that is specified in the CREATE FUNCTION statement.

function implementer. The authorization ID of the owner of the function program and function package.

function package. A package that results from binding the DBRM for a function program.

function package owner. The authorization ID of the user who binds the function program's DBRM into a function package.

function resolution. The process, internal to the DBMS, by which a function invocation is bound to a particular function instance. This process uses the function name, the data types of the arguments, and a list of the applicable schema names (called the *SQL path*) to make the selection. This process is sometimes called *function selection*.

function selection. See *function resolution*.

function signature. The logical concatenation of a fully qualified function name with the data types of all of its parameters.

G

GB. Gigabyte (1 073 741 824 bytes).

GBP. Group buffer pool.

GBP-dependent. The status of a page set or page set partition that is dependent on the group buffer pool. Either read/write interest is active among DB2 subsystems for this page set, or the page set has changed pages in the group buffer pool that have not yet been cast out to disk.

generalized trace facility (GTF). A z/OS service program that records significant system events such as I/O interrupts, SVC interrupts, program interrupts, or external interrupts.

generic resource name. A name that VTAM uses to represent several application programs that provide the same function in order to handle session distribution and balancing in a Sysplex environment.

getpage. An operation in which DB2 accesses a data page.

global lock. A lock that provides concurrency control within and among DB2 subsystems. The scope of the lock is across all DB2 subsystems of a data sharing group.

global lock contention. Conflicts on locking requests between different DB2 members of a data sharing group when those members are trying to serialize shared resources.

governor. See *resource limit facility*.

graphic string. A sequence of DBCS characters.

gross lock. The *shared*, *update*, or *exclusive* mode locks on a table, partition, or table space.

group buffer pool (GBP). A coupling facility cache structure that is used by a data sharing group to cache data and to ensure that the data is consistent for all members.

group buffer pool duplexing. The ability to write data to two instances of a group buffer pool structure: a *primary group buffer pool* and a *secondary group buffer*

pool. z/OS publications refer to these instances as the "old" (for primary) and "new" (for secondary) structures.

group level. The release level of a data sharing group, which is established when the first member migrates to a new release.

group name. The z/OS XCF identifier for a data sharing group.

group restart. A restart of at least one member of a data sharing group after the loss of either locks or the shared communications area.

GTF. Generalized trace facility.

H

handle. In DB2 ODBC, a variable that refers to a data structure and associated resources. See also *statement handle*, *connection handle*, and *environment handle*.

help panel. A screen of information that presents tutorial text to assist a user at the workstation or terminal.

heuristic damage. The inconsistency in data between one or more participants that results when a heuristic decision to resolve an indoubt LUW at one or more participants differs from the decision that is recorded at the coordinator.

heuristic decision. A decision that forces indoubt resolution at a participant by means other than automatic resynchronization between coordinator and participant.

hole. A row of the result table that cannot be accessed because of a delete or an update that has been performed on the row. See also *delete hole* and *update hole*.

home address space. The area of storage that z/OS currently recognizes as *dispatched*.

host. The set of programs and resources that are available on a given TCP/IP instance.

host expression. A Java variable or expression that is referenced by SQL clauses in an SQLJ application program.

host identifier. A name that is declared in the host program.

host language. A programming language in which you can embed SQL statements.

host program. An application program that is written in a host language and that contains embedded SQL statements.

host structure. In an application program, a structure that is referenced by embedded SQL statements.

host variable. In an application program, an application variable that is referenced by embedded SQL statements.

host variable array. An array of elements, each of which corresponds to a value for a column. The dimension of the array determines the maximum number of rows for which the array can be used.

HSM. Hierarchical storage manager.

HTML. Hypertext Markup Language, a standard method for presenting Web data to users.

HTTP. Hypertext Transfer Protocol, a communication protocol that the Web uses.

I

ICF. Integrated catalog facility.

IDCAMS. An IBM program that is used to process access method services commands. It can be invoked as a job or jobstep, from a TSO terminal, or from within a user's application program.

IDCAMS LISTCAT. A facility for obtaining information that is contained in the access method services catalog.

identify. A request that an attachment service program in an address space that is separate from DB2 issues thorough the z/OS subsystem interface to inform DB2 of its existence and to initiate the process of becoming connected to DB2.

identity column. A column that provides a way for DB2 to automatically generate a numeric value for each row. The generated values are unique if cycling is not used. Identity columns are defined with the AS IDENTITY clause. Uniqueness of values can be ensured by defining a unique index that contains only the identity column. A table can have no more than one identity column.

IFCID. Instrumentation facility component identifier.

IFI. Instrumentation facility interface.

IFI call. An invocation of the instrumentation facility interface (IFI) by means of one of its defined functions.

IFP. IMS Fast Path.

image copy. An exact reproduction of all or part of a table space. DB2 provides utility programs to make full image copies (to copy the entire table space) or incremental image copies (to copy only those pages that have been modified since the last image copy).

implied forget. In the presumed-abort protocol, an implied response of *forget* to the second-phase *committed* request from the coordinator. The response is implied when the participant responds to any subsequent request from the coordinator.

IMS. Information Management System.

IMS attachment facility. A DB2 subcomponent that uses z/OS subsystem interface (SSI) protocols and cross-memory linkage to process requests from IMS to DB2 and to coordinate resource commitment.

IMS DB. Information Management System Database.

IMS TM. Information Management System Transaction Manager.

in-abort. A status of a unit of recovery. If DB2 fails after a unit of recovery begins to be rolled back, but before the process is completed, DB2 continues to back out the changes during restart.

in-commit. A status of a unit of recovery. If DB2 fails after beginning its phase 2 commit processing, it "knows," when restarted, that changes made to data are consistent. Such units of recovery are termed *in-commit*.

independent. An object (row, table, or table space) that is neither a parent nor a dependent of another object.

index. A set of pointers that are logically ordered by the values of a key. Indexes can provide faster access to data and can enforce uniqueness on the rows in a table.

index-controlled partitioning. A type of partitioning in which partition boundaries for a partitioned table are controlled by values that are specified on the CREATE INDEX statement. Partition limits are saved in the LIMITKEY column of the SYSIBM.SYSINDEXPART catalog table.

index key. The set of columns in a table that is used to determine the order of index entries.

index partition. A VSAM data set that is contained within a partitioning index space.

index space. A page set that is used to store the entries of one index.

indicator column. A 4-byte value that is stored in a base table in place of a LOB column.

indicator variable. A variable that is used to represent the null value in an application program. If the value for the selected column is null, a negative value is placed in the indicator variable.

indoubt. A status of a unit of recovery. If DB2 fails after it has finished its phase 1 commit processing and before it has started phase 2, only the commit coordinator knows if an individual unit of recovery is to be committed or rolled back. At emergency restart, if DB2 lacks the information it needs to make this decision, the status of the unit of recovery is *indoubt* until DB2 obtains this information from the coordinator. More than one unit of recovery can be indoubt at restart.

indoubt resolution. The process of resolving the status of an indoubt logical unit of work to either the committed or the rollback state.

inflight. A status of a unit of recovery. If DB2 fails before its unit of recovery completes phase 1 of the commit process, it merely backs out the updates of its unit of recovery at restart. These units of recovery are termed *inflight*.

inheritance. The passing downstream of class resources or attributes from a parent class in the class hierarchy to a child class.

initialization file. For DB2 ODBC applications, a file containing values that can be set to adjust the performance of the database manager.

inline copy. A copy that is produced by the LOAD or REORG utility. The data set that the inline copy produces is logically equivalent to a full image copy that is produced by running the COPY utility with read-only access (SHRLEVEL REFERENCE).

inner join. The result of a join operation that includes only the matched rows of both tables that are being joined. See also *join*.

inoperative package. A package that cannot be used because one or more user-defined functions or procedures that the package depends on were dropped. Such a package must be explicitly rebound. Contrast with *invalid package.*

insensitive cursor. A cursor that is not sensitive to inserts, updates, or deletes that are made to the underlying rows of a result table after the result table has been materialized.

insert trigger. A trigger that is defined with the triggering SQL operation INSERT.

install. The process of preparing a DB2 subsystem to operate as a z/OS subsystem.

installation verification scenario. A sequence of operations that exercises the main DB2 functions and tests whether DB2 was correctly installed.

instrumentation facility component identifier (IFCID). A value that names and identifies a trace record of an event that can be traced. As a parameter on the START TRACE and MODIFY TRACE commands, it specifies that the corresponding event is to be traced.

instrumentation facility interface (IFI). A programming interface that enables programs to obtain online trace data about DB2, to submit DB2 commands, and to pass data to DB2.

Interactive System Productivity Facility (ISPF). An IBM licensed program that provides interactive dialog services in a z/OS environment.

inter-DB2 R/W interest. A property of data in a table space, index, or partition that has been opened by more than one member of a data sharing group and that has been opened for writing by at least one of those members.

intermediate database server. The target of a request from a local application or a remote application requester that is forwarded to another database server. In the DB2 environment, the remote request is forwarded transparently to another database server if the object that is referenced by a three-part name does not reference the local location.

internationalization. The support for an encoding scheme that is able to represent the code points of characters from many different geographies and languages. To support all geographies, the Unicode standard requires more than 1 byte to represent a single character. See also *Unicode*.

internal resource lock manager (IRLM). A z/OS subsystem that DB2 uses to control communication and database locking.

International Organization for Standardization. An international body charged with creating standards to facilitate the exchange of goods and services as well as cooperation in intellectual, scientific, technological, and economic activity.

invalid package. A package that depends on an object (other than a user-defined function) that is dropped. Such a package is implicitly rebound on invocation. Contrast with *inoperative package.*

invariant character set. (1) A character set, such as the syntactic character set, whose code point assignments do not change from code page to code page. (2) A minimum set of characters that is available as part of all character sets.

IP address. A 4-byte value that uniquely identifies a TCP/IP host.

IRLM. Internal resource lock manager.

ISO. International Organization for Standardization.

isolation level. The degree to which a unit of work is isolated from the updating operations of other units of work. See also *cursor stability*, *read stability*, *repeatable read*, and *uncommitted read*.

ISPF. Interactive System Productivity Facility.

ISPF/PDF. Interactive System Productivity Facility/Program Development Facility.

iterator. In SQLJ, an object that contains the result set of a query. An iterator is equivalent to a cursor in other host languages.

iterator declaration clause. In SQLJ, a statement that generates an iterator declaration class. An iterator is an object of an iterator declaration class.

J

Japanese Industrial Standard. An encoding scheme that is used to process Japanese characters.

JAR. Java Archive.

Java Archive (JAR). A file format that is used for aggregating many files into a single file.

JCL. Job control language.

JDBC. A Sun Microsystems database application programming interface (API) for Java that allows programs to access database management systems by using callable SQL. JDBC does not require the use of an SQL preprocessor. In addition, JDBC provides an architecture that lets users add modules called *database drivers*, which link the application to their choice of database management systems at run time.

JES. Job Entry Subsystem.

JIS. Japanese Industrial Standard.

job control language (JCL). A control language that is used to identify a job to an operating system and to describe the job's requirements.

Job Entry Subsystem (JES). An IBM licensed program that receives jobs into the system and processes all output data that is produced by the jobs.

join. A relational operation that allows retrieval of data from two or more tables based on matching column values. See also *equijoin*, *full outer join*, *inner join*, *left outer join*, *outer join*, *and right outer join*.

K

KB. Kilobyte (1024 bytes).

Kerberos. A network authentication protocol that is designed to provide strong authentication for client/server applications by using secret-key cryptography.

Kerberos ticket. A transparent application mechanism that transmits the identity of an initiating principal to its target. A simple ticket contains the principal's

identity, a session key, a timestamp, and other information, which is sealed using the target's secret key.

key. A column or an ordered collection of columns that is identified in the description of a table, index, or referential constraint. The same column can be part of more than one key.

key-sequenced data set (KSDS). A VSAM file or data set whose records are loaded in key sequence and controlled by an index.

keyword. In SQL, a name that identifies an option that is used in an SQL statement.

KSDS. Key-sequenced data set.

L

labeled duration. A number that represents a duration of years, months, days, hours, minutes, seconds, or microseconds.

large object (LOB). A sequence of bytes representing bit data, single-byte characters, double-byte characters, or a mixture of single- and double-byte characters. A LOB can be up to 2 GB–1 byte in length. See also *BLOB*, *CLOB*, and *DBCLOB*.

last agent optimization. An optimized commit flow for either presumed-nothing or presumed-abort protocols in which the last agent, or final participant, becomes the commit coordinator. This flow saves at least one message.

latch. A DB2 internal mechanism for controlling concurrent events or the use of system resources.

LCID. Log control interval definition.

LDS. Linear data set.

leaf page. A page that contains pairs of keys and RIDs and that points to actual data. Contrast with *nonleaf page*.

left outer join. The result of a join operation that includes the matched rows of both tables that are being joined, and that preserves the unmatched rows of the first table. See also *join*.

limit key. The highest value of the index key for a partition.

linear data set (LDS). A VSAM data set that contains data but no control information. A linear data set can be accessed as a byte-addressable string in virtual storage.

linkage editor. A computer program for creating load modules from one or more object modules or load modules by resolving cross references among the modules and, if necessary, adjusting addresses.

link-edit. The action of creating a loadable computer program using a linkage editor.

list. A type of object, which DB2 utilities can process, that identifies multiple table spaces, multiple index spaces, or both. A list is defined with the LISTDEF utility control statement.

list structure. A coupling facility structure that lets data be shared and manipulated as elements of a queue.

LLE. Load list element.

L-lock. Logical lock.

load list element. A z/OS control block that controls the loading and deleting of a particular load module based on entry point names.

load module. A program unit that is suitable for loading into main storage for execution. The output of a linkage editor.

LOB. Large object.

LOB locator. A mechanism that allows an application program to manipulate a large object value in the database system. A LOB locator is a fullword integer value that represents a single LOB value. An application program retrieves a LOB locator into a host variable and can then apply SQL operations to the associated LOB value using the locator.

LOB lock. A lock on a LOB value.

LOB table space. A table space in an auxiliary table that contains all the data for a particular LOB column in the related base table.

local. A way of referring to any object that the local DB2 subsystem maintains. A *local table*, for example, is a table that is maintained by the local DB2 subsystem. Contrast with *remote*.

locale. The definition of a subset of a user's environment that combines a CCSID and characters that are defined for a specific language and country.

local lock. A lock that provides intra-DB2 concurrency control, but not inter-DB2 concurrency control; that is, its scope is a single DB2.

local subsystem. The unique relational DBMS to which the user or application program is directly connected (in the case of DB2, by one of the DB2 attachment facilities).

location. The unique name of a database server. An application uses the location name to access a DB2

database server. A database alias can be used to override the location name when accessing a remote server.

location alias. Another name by which a database server identifies itself in the network. Applications can use this name to access a DB2 database server.

lock. A means of controlling concurrent events or access to data. DB2 locking is performed by the IRLM.

lock duration. The interval over which a DB2 lock is held.

lock escalation. The promotion of a lock from a row, page, or LOB lock to a table space lock because the number of page locks that are concurrently held on a given resource exceeds a preset limit.

locking. The process by which the integrity of data is ensured. Locking prevents concurrent users from accessing inconsistent data.

lock mode. A representation for the type of access that concurrently running programs can have to a resource that a DB2 lock is holding.

lock object. The resource that is controlled by a DB2 lock.

lock promotion. The process of changing the size or mode of a DB2 lock to a higher, more restrictive level.

lock size. The amount of data that is controlled by a DB2 lock on table data; the value can be a row, a page, a LOB, a partition, a table, or a table space.

lock structure. A coupling facility data structure that is composed of a series of lock entries to support shared and exclusive locking for logical resources.

log. A collection of records that describe the events that occur during DB2 execution and that indicate their sequence. The information thus recorded is used for recovery in the event of a failure during DB2 execution.

log control interval definition. A suffix of the physical log record that tells how record segments are placed in the physical control interval.

logical claim. A claim on a logical partition of a nonpartitioning index.

logical data modeling. The process of documenting the comprehensive business information requirements in an accurate and consistent format. Data modeling is the first task of designing a database.

logical drain. A drain on a logical partition of a nonpartitioning index.

logical index partition. The set of all keys that reference the same data partition.

logical lock (L-lock). The lock type that transactions use to control intra- and inter-DB2 data concurrency between transactions. Contrast with *physical lock (P-lock)*.

logically complete. A state in which the concurrent copy process is finished with the initialization of the target objects that are being copied. The target objects are available for update.

logical page list (LPL). A list of pages that are in error and that cannot be referenced by applications until the pages are recovered. The page is in *logical error* because the actual media (coupling facility or disk) might not contain any errors. Usually a connection to the media has been lost.

logical partition. A set of key or RID pairs in a nonpartitioning index that are associated with a particular partition.

logical recovery pending (LRECP). The state in which the data and the index keys that reference the data are inconsistent.

logical unit (LU). An access point through which an application program accesses the SNA network in order to communicate with another application program.

logical unit of work (LUW). The processing that a program performs between synchronization points.

logical unit of work identifier (LUWID). A name that uniquely identifies a thread within a network. This name consists of a fully-qualified LU network name, an LUW instance number, and an LUW sequence number.

log initialization. The first phase of restart processing during which DB2 attempts to locate the current end of the log.

log record header (LRH). A prefix, in every logical record, that contains control information.

log record sequence number (LRSN). A unique identifier for a log record that is associated with a data sharing member. DB2 uses the LRSN for recovery in the data sharing environment.

log truncation. A process by which an explicit starting RBA is established. This RBA is the point at which the next byte of log data is to be written.

LPL. Logical page list.

LRECP. Logical recovery pending.

LRH. Log record header.

LRSN. Log record sequence number.

LU. Logical unit.

LU name. Logical unit name, which is the name by which VTAM refers to a node in a network. Contrast with *location name*.

LUW. Logical unit of work.

LUWID. Logical unit of work identifier.

M

mapping table. A table that the REORG utility uses to map the associations of the RIDs of data records in the original copy and in the shadow copy. This table is created by the user.

mass delete. The deletion of all rows of a table.

master terminal. The IMS logical terminal that has complete control of IMS resources during online operations.

master terminal operator (MTO). See *master terminal.*

materialize. (1) The process of putting rows from a view or nested table expression into a work file for additional processing by a query.

(2) The placement of a LOB value into contiguous storage. Because LOB values can be very large, DB2 avoids materializing LOB data until doing so becomes absolutely necessary.

| **materialized query table.** A table that is used to
| contain information that is derived and can be
| summarized from one or more source tables.

MB. Megabyte (1 048 576 bytes).

MBCS. Multibyte character set. UTF-8 is an example of an MBCS. Characters in UTF-8 can range from 1 to 4 bytes in DB2.

member name. The z/OS XCF identifier for a particular DB2 subsystem in a data sharing group.

menu. A displayed list of available functions for selection by the operator. A menu is sometimes called a *menu panel*.

metalanguage. A language that is used to create other specialized languages.

migration. The process of converting a subsystem with a previous release of DB2 to an updated or current release. In this process, you can acquire the functions of the updated or current release without losing the data that you created on the previous release.

mixed data string. A character string that can contain both single-byte and double-byte characters.

MLPA. Modified link pack area.

MODEENT. A VTAM macro instruction that associates a logon mode name with a set of parameters representing session protocols. A set of MODEENT macro instructions defines a logon mode table.

modeling database. A DB2 database that you create on your workstation that you use to model a DB2 UDB for z/OS subsystem, which can then be evaluated by the Index Advisor.

mode name. A VTAM name for the collection of physical and logical characteristics and attributes of a session.

modify locks. An L-lock or P-lock with a MODIFY attribute. A list of these active locks is kept at all times in the coupling facility lock structure. If the requesting DB2 subsystem fails, that DB2 subsystem's modify locks are converted to retained locks.

MPP. Message processing program (in IMS).

MTO. Master terminal operator.

multibyte character set (MBCS). A character set that represents single characters with more than a single byte. Contrast with *single-byte character set* and *double-byte character set*. See also *Unicode*.

multidimensional analysis. The process of assessing and evaluating an enterprise on more than one level.

Multiple Virtual Storage. An element of the z/OS operating system. This element is also called the Base Control Program (BCP).

multisite update. Distributed relational database processing in which data is updated in more than one location within a single unit of work.

multithreading. Multiple TCBs that are executing one copy of DB2 ODBC code concurrently (sharing a processor) or in parallel (on separate central processors).

must-complete. A state during DB2 processing in which the entire operation must be completed to maintain data integrity.

mutex. Pthread mutual exclusion; a lock. A Pthread mutex variable is used as a locking mechanism to allow serialization of critical sections of code by temporarily blocking the execution of all but one thread.

| **MVS.** See *Multiple Virtual Storage*.

N

negotiable lock. A lock whose mode can be downgraded, by agreement among contending users, to be compatible to all. A physical lock is an example of a negotiable lock.

nested table expression. A fullselect in a FROM clause (surrounded by parentheses).

network identifier (NID). The network ID that is assigned by IMS or CICS, or if the connection type is RRSAF, the RRS unit of recovery ID (URID).

NID. Network identifier.

nonleaf page. A page that contains keys and page numbers of other pages in the index (either leaf or nonleaf pages). Nonleaf pages never point to actual data.

nonpartitioned index. An index that is not physically partitioned. Both partitioning indexes and secondary indexes can be nonpartitioned.

nonscrollable cursor. A cursor that can be moved only in a forward direction. Nonscrollable cursors are sometimes called forward-only cursors or serial cursors.

normalization. A key step in the task of building a logical relational database design. Normalization helps you avoid redundancies and inconsistencies in your data. An entity is normalized if it meets a set of constraints for a particular normal form (first normal form, second normal form, and so on). Contrast with *denormalization*.

nondeterministic function. A user-defined function whose result is not solely dependent on the values of the input arguments. That is, successive invocations with the same argument values can produce a different answer. this type of function is sometimes called a *variant* function. Contrast this with a *deterministic function* (sometimes called a *not-variant function*), which always produces the same result for the same inputs.

not-variant function. See *deterministic function*.

NPSI. See *nonpartitioned secondary index.*

NRE. Network recovery element.

NUL. The null character ('\0'), which is represented by the value X'00'. In C, this character denotes the end of a string.

null. A special value that indicates the absence of information.

NULLIF. A scalar function that evaluates two passed expressions, returning either NULL if the arguments are equal or the value of the first argument if they are not.

null-terminated host variable. A varying-length host variable in which the end of the data is indicated by a null terminator.

null terminator. In C, the value that indicates the end of a string. For EBCDIC, ASCII, and Unicode UTF-8 strings, the null terminator is a single-byte value (X'00'). For Unicode UCS-2 (wide) strings, the null terminator is a double-byte value (X'0000').

O

OASN (origin application schedule number). In IMS, a 4-byte number that is assigned sequentially to each IMS schedule since the last cold start of IMS. The OASN is used as an identifier for a unit of work. In an 8-byte format, the first 4 bytes contain the schedule number and the last 4 bytes contain the number of IMS sync points (*commit points*) during the current schedule. The OASN is part of the NID for an IMS connection.

ODBC. Open Database Connectivity.

ODBC driver. A dynamically-linked library (DLL) that implements ODBC function calls and interacts with a data source.

OBID. Data object identifier.

Open Database Connectivity (ODBC). A Microsoft® database application programming interface (API) for C that allows access to database management systems by using callable SQL. ODBC does not require the use of an SQL preprocessor. In addition, ODBC provides an architecture that lets users add modules called *database drivers*, which link the application to their choice of database management systems at run time. This means that applications no longer need to be directly linked to the modules of all the database management systems that are supported.

ordinary identifier. An uppercase letter followed by zero or more characters, each of which is an uppercase letter, a digit, or the underscore character. An ordinary identifier must not be a reserved word.

ordinary token. A numeric constant, an ordinary identifier, a host identifier, or a keyword.

originating task. In a parallel group, the primary agent that receives data from other execution units (referred to as *parallel tasks*) that are executing portions of the query in parallel.

OS/390. Operating System/390®.

outer join. The result of a join operation that includes the matched rows of both tables that are being joined and preserves some or all of the unmatched rows of the tables that are being joined. See also *join*.

overloaded function. A function name for which multiple function instances exist.

P

package. An object containing a set of SQL statements that have been statically bound and that is available for processing. A package is sometimes also called an *application package*.

package list. An ordered list of package names that may be used to extend an application plan.

package name. The name of an object that is created by a BIND PACKAGE or REBIND PACKAGE command. The object is a bound version of a database request module (DBRM). The name consists of a location name, a collection ID, a package ID, and a version ID.

page. A unit of storage within a table space (4 KB, 8 KB, 16 KB, or 32 KB) or index space (4 KB). In a table space, a page contains one or more rows of a table. In a LOB table space, a LOB value can span more than one page, but no more than one LOB value is stored on a page.

page set. Another way to refer to a table space or index space. Each page set consists of a collection of VSAM data sets.

page set recovery pending (PSRCP). A restrictive state of an index space. In this case, the entire page set must be recovered. Recovery of a logical part is prohibited.

panel. A predefined display image that defines the locations and characteristics of display fields on a display surface (for example, a *menu panel*).

parallel complex. A cluster of machines that work together to handle multiple transactions and applications.

parallel group. A set of consecutive operations that execute in parallel and that have the same number of parallel tasks.

parallel I/O processing. A form of I/O processing in which DB2 initiates multiple concurrent requests for a single user query and performs I/O processing concurrently (in *parallel*) on multiple data partitions.

parallelism assistant. In Sysplex query parallelism, a DB2 subsystem that helps to process parts of a parallel query that originates on another DB2 subsystem in the data sharing group.

parallelism coordinator. In Sysplex query parallelism, the DB2 subsystem from which the parallel query originates.

Parallel Sysplex. A set of z/OS systems that communicate and cooperate with each other through certain multisystem hardware components and software services to process customer workloads.

parallel task. The execution unit that is dynamically created to process a query in parallel. A parallel task is implemented by a z/OS service request block.

parameter marker. A question mark (?) that appears in a statement string of a dynamic SQL statement. The question mark can appear where a host variable could appear if the statement string were a static SQL statement.

parameter-name. An SQL identifier that designates a parameter in an SQL procedure or an SQL function.

parent key. A primary key or unique key in the parent table of a referential constraint. The values of a parent key determine the valid values of the foreign key in the referential constraint.

parent lock. For explicit hierarchical locking, a lock that is held on a resource that might have child locks that are lower in the hierarchy. A parent lock is usually the table space lock or the partition intent lock. See also *child lock*.

parent row. A row whose primary key value is the foreign key value of a dependent row.

parent table. A table whose primary key is referenced by the foreign key of a dependent table.

parent table space. A table space that contains a parent table. A table space containing a dependent of that table is a dependent table space.

participant. An entity other than the commit coordinator that takes part in the commit process. The term participant is synonymous with *agent* in SNA.

partition. A portion of a page set. Each partition corresponds to a single, independently extendable data set. Partitions can be extended to a maximum size of 1, 2, or 4 GB, depending on the number of partitions in the partitioned page set. All partitions of a given page set have the same maximum size.

partitioned data set (PDS). A data set in disk storage that is divided into partitions, which are called members. Each partition can contain a program, part of a program, or data. The term partitioned data set is synonymous with program library.

partitioned index. An index that is physically partitioned. Both partitioning indexes and secondary indexes can be partitioned.

partitioned page set. A partitioned table space or an index space. Header pages, space map pages, data pages, and index pages reference data only within the scope of the partition.

partitioned table space. A table space that is subdivided into parts (based on index key range), each of which can be processed independently by utilities.

partitioning index. An index in which the leftmost columns are the partitioning columns of the table. The index can be partitioned or nonpartitioned.

partition pruning. The removal from consideration of inapplicable partitions through setting up predicates in a query on a partitioned table to access only certain partitions to satisfy the query.

partner logical unit. An access point in the SNA network that is connected to the local DB2 subsystem by way of a VTAM conversation.

path. See *SQL path*.

PCT. Program control table (in CICS).

PDS. Partitioned data set.

piece. A data set of a nonpartitioned page set.

physical claim. A claim on an entire nonpartitioning index.

physical consistency. The state of a page that is not in a partially changed state.

physical drain. A drain on an entire nonpartitioning index.

physical lock (P-lock). A type of lock that DB2 acquires to provide consistency of data that is cached in different DB2 subsystems. Physical locks are used only in data sharing environments. Contrast with *logical lock (L-lock)*.

physical lock contention. Conflicting states of the requesters for a physical lock. See also *negotiable lock*.

physically complete. The state in which the concurrent copy process is completed and the output data set has been created.

plan. See *application plan*.

plan allocation. The process of allocating DB2 resources to a plan in preparation for execution.

plan member. The bound copy of a DBRM that is identified in the member clause.

plan name. The name of an application plan.

plan segmentation. The dividing of each plan into sections. When a section is needed, it is independently brought into the EDM pool.

P-lock. Physical lock.

PLT. Program list table (in CICS).

point of consistency. A time when all recoverable data that an application accesses is consistent with other data. The term point of consistency is synonymous with *sync point* or *commit point*.

policy. See *CFRM policy*.

Portable Operating System Interface (POSIX). The IEEE operating system interface standard, which defines the Pthread standard of threading. See also *Pthread*.

POSIX. Portable Operating System Interface.

postponed abort UR. A unit of recovery that was inflight or in-abort, was interrupted by system failure or cancellation, and did not complete backout during restart.

PPT. (1) Processing program table (in CICS). (2) Program properties table (in z/OS).

precision. In SQL, the total number of digits in a decimal number (called the *size* in the C language). In the C language, the number of digits to the right of the decimal point (called the *scale* in SQL). The DB2 library uses the SQL terms.

precompilation. A processing of application programs containing SQL statements that takes place before compilation. SQL statements are replaced with statements that are recognized by the host language compiler. Output from this precompilation includes source code that can be submitted to the compiler and the database request module (DBRM) that is input to the bind process.

predicate. An element of a search condition that expresses or implies a comparison operation.

prefix. A code at the beginning of a message or record.

preformat. The process of preparing a VSAM ESDS for DB2 use, by writing specific data patterns.

prepare. The first phase of a two-phase commit process in which all participants are requested to prepare for commit.

prepared SQL statement. A named object that is the executable form of an SQL statement that has been processed by the PREPARE statement.

presumed-abort. An optimization of the presumed-nothing two-phase commit protocol that reduces the number of recovery log records, the duration of state maintenance, and the number of messages between coordinator and participant. The optimization also modifies the indoubt resolution responsibility.

presumed-nothing. The standard two-phase commit protocol that defines coordinator and participant responsibilities, relative to logical unit of work states, recovery logging, and indoubt resolution.

primary authorization ID. The authorization ID that is used to identify the application process to DB2.

primary group buffer pool. For a duplexed group buffer pool, the structure that is used to maintain the coherency of cached data. This structure is used for page registration and cross-invalidation. The z/OS equivalent is *old* structure. Compare with *secondary group buffer pool*.

primary index. An index that enforces the uniqueness of a primary key.

primary key. In a relational database, a unique, nonnull key that is part of the definition of a table. A table cannot be defined as a parent unless it has a unique key or primary key.

principal. An entity that can communicate securely with another entity. In Kerberos, principals are represented as entries in the Kerberos registry database and include users, servers, computers, and others.

principal name. The name by which a principal is known to the DCE security services.

private connection. A communications connection that is specific to DB2.

private protocol access. A method of accessing distributed data by which you can direct a query to another DB2 system. Contrast with *DRDA access*.

private protocol connection. A DB2 private connection of the application process. See also *private connection*.

privilege. The capability of performing a specific function, sometimes on a specific object. The types of privileges are:

explicit privileges, which have names and are held as the result of SQL GRANT and REVOKE statements. For example, the SELECT privilege.
implicit privileges, which accompany the ownership of an object, such as the privilege to drop a synonym that one owns, or the holding of an authority, such as the privilege of SYSADM authority to terminate any utility job.

privilege set. For the installation SYSADM ID, the set of all possible privileges. For any other authorization ID, the set of all privileges that are recorded for that ID in the DB2 catalog.

process. In DB2, the unit to which DB2 allocates resources and locks. Sometimes called an *application process*, a process involves the execution of one or more programs. The execution of an SQL statement is always associated with some process. The means of initiating and terminating a process are dependent on the environment.

program. A single, compilable collection of executable statements in a programming language.

program temporary fix (PTF). A solution or bypass of a problem that is diagnosed as a result of a defect in a current unaltered release of a licensed program. An authorized program analysis report (APAR) fix is corrective service for an existing problem. A PTF is preventive service for problems that might be encountered by other users of the product. A PTF is *temporary*, because a permanent fix is usually not incorporated into the product until its next release.

protected conversation. A VTAM conversation that supports two-phase commit flows.

PSRCP. Page set recovery pending.

PTF. Program temporary fix.

Pthread. The POSIX threading standard model for splitting an application into subtasks. The Pthread standard includes functions for creating threads, terminating threads, synchronizing threads through locking, and other thread control facilities.

Q

QMF™. Query Management Facility.

QSAM. Queued sequential access method.

query. A component of certain SQL statements that specifies a result table.

query block. The part of a query that is represented by one of the FROM clauses. Each FROM clause can have multiple query blocks, depending on DB2's internal processing of the query.

query CP parallelism. Parallel execution of a single query, which is accomplished by using multiple tasks. See also *Sysplex query parallelism*.

query I/O parallelism. Parallel access of data, which is accomplished by triggering multiple I/O requests within a single query.

queued sequential access method (QSAM). An extended version of the basic sequential access method (BSAM). When this method is used, a queue of data blocks is formed. Input data blocks await processing, and output data blocks await transfer to auxiliary storage or to an output device.

quiesce point. A point at which data is consistent as a result of running the DB2 QUIESCE utility.

quiesced member state. A state of a member of a data sharing group. An active member becomes quiesced when a STOP DB2 command takes effect without a failure. If the member's task, address space, or z/OS system fails before the command takes effect, the member state is failed.

R

RACF. Resource Access Control Facility, which is a component of the z/OS Security Server.

RAMAC®. IBM family of enterprise disk storage system products.

RBA. Relative byte address.

RCT. Resource control table (in CICS attachment facility).

RDB. Relational database.

RDBMS. Relational database management system.

RDBNAM. Relational database name.

RDF. Record definition field.

read stability (RS). An isolation level that is similar to repeatable read but does not completely isolate an application process from all other concurrently executing application processes. Under level RS, an application that issues the same query more than once might read additional rows that were inserted and committed by a concurrently executing application process.

rebind. The creation of a new application plan for an application program that has been bound previously. If, for example, you have added an index for a table that your application accesses, you must rebind the application in order to take advantage of that index.

rebuild. The process of reallocating a coupling facility structure. For the shared communications area (SCA) and lock structure, the structure is repopulated; for the group buffer pool, changed pages are usually cast out to disk, and the new structure is populated only with changed pages that were not successfully cast out.

RECFM. Record format.

record. The storage representation of a row or other data.

record identifier (RID). A unique identifier that DB2 uses internally to identify a row of data in a table. Compare with *row ID*.

record identifier (RID) pool. An area of main storage that is used for sorting record identifiers during list-prefetch processing.

record length. The sum of the length of all the columns in a table, which is the length of the data as it is physically stored in the database. Records can be fixed length or varying length, depending on how the columns are defined. If all columns are fixed-length columns, the record is a fixed-length record. If one or more columns are varying-length columns, the record is a varying-length column.

Recoverable Resource Manager Services attachment facility (RRSAF). A DB2 subcomponent that uses Resource Recovery Services to coordinate resource commitment between DB2 and all other resource managers that also use RRS in a z/OS system.

recovery. The process of rebuilding databases after a system failure.

recovery log. A collection of records that describes the events that occur during DB2 execution and indicates their sequence. The recorded information is used for recovery in the event of a failure during DB2 execution.

recovery manager. (1) A subcomponent that supplies coordination services that control the interaction of DB2 resource managers during commit, abort, checkpoint, and restart processes. The recovery manager also supports the recovery mechanisms of other subsystems (for example, IMS) by acting as a participant in the other subsystem's process for protecting data that has reached a point of consistency. (2) A coordinator or a participant (or both), in the execution of a two-phase commit, that can access a recovery log that maintains the state of the logical unit of work and names the immediate upstream coordinator and downstream participants.

recovery pending (RECP). A condition that prevents SQL access to a table space that needs to be recovered.

recovery token. An identifier for an element that is used in recovery (for example, NID or URID).

RECP. Recovery pending.

redo. A state of a unit of recovery that indicates that changes are to be reapplied to the disk media to ensure data integrity.

reentrant. Executable code that can reside in storage as one shared copy for all threads. Reentrant code is not self-modifying and provides separate storage areas for each thread. Reentrancy is a compiler and operating system concept, and reentrancy alone is not enough to guarantee logically consistent results when multithreading. See also *threadsafe*.

referential constraint. The requirement that nonnull values of a designated foreign key are valid only if they equal values of the primary key of a designated table.

referential integrity. The state of a database in which all values of all foreign keys are valid. Maintaining referential integrity requires the enforcement of referential constraints on all operations that change the data in a table on which the referential constraints are defined.

referential structure. A set of tables and relationships that includes at least one table and, for every table in the set, all the relationships in which that table participates and all the tables to which it is related.

refresh age. The time duration between the current time and the time during which a materialized query table was last refreshed.

registry. See *registry database*.

registry database. A database of security information about principals, groups, organizations, accounts, and security policies.

relational database (RDB). A database that can be perceived as a set of tables and manipulated in accordance with the relational model of data.

relational database management system (RDBMS). A collection of hardware and software that organizes and provides access to a relational database.

relational database name (RDBNAM). A unique identifier for an RDBMS within a network. In DB2, this must be the value in the LOCATION column of table SYSIBM.LOCATIONS in the CDB. DB2 publications refer to the name of another RDBMS as a LOCATION value or a location name.

relationship. A defined connection between the rows of a table or the rows of two tables. A relationship is the internal representation of a referential constraint.

relative byte address (RBA). The offset of a data record or control interval from the beginning of the storage space that is allocated to the data set or file to which it belongs.

remigration. The process of returning to a current release of DB2 following a fallback to a previous release. This procedure constitutes another migration process.

remote. Any object that is maintained by a remote DB2 subsystem (that is, by a DB2 subsystem other than the local one). A *remote view*, for example, is a view that is maintained by a remote DB2 subsystem. Contrast with *local*.

remote attach request. A request by a remote location to attach to the local DB2 subsystem. Specifically, the request that is sent is an SNA Function Management Header 5.

remote subsystem. Any relational DBMS, except the *local subsystem*, with which the user or application can communicate. The subsystem need not be remote in any physical sense, and might even operate on the same processor under the same z/OS system.

reoptimization. The DB2 process of reconsidering the access path of an SQL statement at run time; during reoptimization, DB2 uses the values of host variables, parameter markers, or special registers.

REORG pending (REORP). A condition that restricts SQL access and most utility access to an object that must be reorganized.

REORP. REORG pending.

repeatable read (RR). The isolation level that provides maximum protection from other executing application programs. When an application program executes with repeatable read protection, rows that the program references cannot be changed by other programs until the program reaches a commit point.

repeating group. A situation in which an entity includes multiple attributes that are inherently the same. The presence of a repeating group violates the requirement of first normal form. In an entity that satisfies the requirement of first normal form, each attribute is independent and unique in its meaning and its name. See also *normalization*.

replay detection mechanism. A method that allows a principal to detect whether a request is a valid request from a source that can be trusted or whether an untrustworthy entity has captured information from a previous exchange and is replaying the information exchange to gain access to the principal.

request commit. The vote that is submitted to the prepare phase if the participant has modified data and is prepared to commit or roll back.

requester. The source of a request to access data at a remote server. In the DB2 environment, the requester function is provided by the distributed data facility.

resource. The object of a lock or claim, which could be a table space, an index space, a data partition, an index partition, or a logical partition.

resource allocation. The part of plan allocation that deals specifically with the database resources.

resource control table (RCT). A construct of the CICS attachment facility, created by site-provided macro parameters, that defines authorization and access attributes for transactions or transaction groups.

resource definition online. A CICS feature that you use to define CICS resources online without assembling tables.

resource limit facility (RLF). A portion of DB2 code that prevents dynamic manipulative SQL statements from exceeding specified time limits. The resource limit facility is sometimes called the governor.

resource limit specification table (RLST). A site-defined table that specifies the limits to be enforced by the resource limit facility.

resource manager. (1) A function that is responsible for managing a particular resource and that guarantees the consistency of all updates made to recoverable resources within a logical unit of work. The resource that is being managed can be physical (for example, disk or main storage) or logical (for example, a particular type of system service). (2) A participant, in the execution of a two-phase commit, that has recoverable resources that could have been modified. The resource manager has access to a recovery log so that it can commit or roll back the effects of the logical unit of work to the recoverable resources.

restart pending (RESTP). A restrictive state of a page set or partition that indicates that restart (backout) work needs to be performed on the object. All access to the page set or partition is denied except for access by the:
- RECOVER POSTPONED command
- Automatic online backout (which DB2 invokes after restart if the system parameter LBACKOUT=AUTO)

RESTP. Restart pending.

result set. The set of rows that a stored procedure returns to a client application.

result set locator. A 4-byte value that DB2 uses to uniquely identify a query result set that a stored procedure returns.

result table. The set of rows that are specified by a SELECT statement.

retained lock. A MODIFY lock that a DB2 subsystem was holding at the time of a subsystem failure. The lock is retained in the coupling facility lock structure across a DB2 failure.

RID. Record identifier.

RID pool. Record identifier pool.

right outer join. The result of a join operation that includes the matched rows of both tables that are being joined and preserves the unmatched rows of the second join operand. See also *join*.

RLF. Resource limit facility.

RLST. Resource limit specification table.

RMID. Resource manager identifier.

RO. Read-only access.

rollback. The process of restoring data that was changed by SQL statements to the state at its last commit point. All locks are freed. Contrast with *commit*.

root page. The index page that is at the highest level (or the beginning point) in an index.

routine. A term that refers to either a user-defined function or a stored procedure.

row. The horizontal component of a table. A row consists of a sequence of values, one for each column of the table.

ROWID. Row identifier.

row identifier (ROWID). A value that uniquely identifies a row. This value is stored with the row and never changes.

row lock. A lock on a single row of data.

| **rowset.** A set of rows for which a cursor position is
| established.

rowset cursor. A cursor that is defined so that one or more rows can be returned as a rowset for a single FETCH statement, and the cursor is positioned on the set of rows that is fetched.

rowset-positioned access. The ability to retrieve multiple rows from a single FETCH statement.

| **row-positioned access.** The ability to retrieve a single
| row from a single FETCH statement.

row trigger. A trigger that is defined with the trigger granularity FOR EACH ROW.

RRE. Residual recovery entry (in IMS).

RRSAF. Recoverable Resource Manager Services attachment facility.

RS. Read stability.

RTT. Resource translation table.

RURE. Restart URE.

S

savepoint. A named entity that represents the state of data and schemas at a particular point in time within a unit of work. SQL statements exist to set a savepoint, release a savepoint, and restore data and schemas to the state that the savepoint represents. The restoration of data and schemas to a savepoint is usually referred to as *rolling back to a savepoint*.

SBCS. Single-byte character set.

SCA. Shared communications area.

scalar function. An SQL operation that produces a
single value from another value and is expressed as a
function name, followed by a list of arguments that are
enclosed in parentheses. Contrast with *aggregate*
function.

scale. In SQL, the number of digits to the right of the decimal point (called the *precision* in the C language). The DB2 library uses the SQL definition.

schema. (1) The organization or structure of a database. (2) A logical grouping for user-defined functions, distinct types, triggers, and stored procedures. When an object of one of these types is created, it is assigned to one schema, which is determined by the name of the object. For example, the following statement creates a distinct type T in schema C:

```
CREATE DISTINCT TYPE C.T ...
```

scrollability. The ability to use a cursor to fetch in either a forward or backward direction. The FETCH statement supports multiple fetch orientations to indicate the new position of the cursor. See also *fetch orientation*.

scrollable cursor. A cursor that can be moved in both a forward and a backward direction.

SDWA. System diagnostic work area.

search condition. A criterion for selecting rows from a table. A search condition consists of one or more predicates.

secondary authorization ID. An authorization ID that has been associated with a primary authorization ID by an authorization exit routine.

secondary group buffer pool. For a duplexed group buffer pool, the structure that is used to back up changed pages that are written to the primary group buffer pool. No page registration or cross-invalidation occurs using the secondary group buffer pool. The z/OS equivalent is *new* structure.

secondary index. A nonpartitioning index on a partitioned table.

section. The segment of a plan or package that contains the executable structures for a single SQL statement. For most SQL statements, one section in the plan exists for each SQL statement in the source program. However, for cursor-related statements, the DECLARE, OPEN, FETCH, and CLOSE statements reference the same section because they each refer to the SELECT statement that is named in the DECLARE CURSOR statement. SQL statements such as COMMIT, ROLLBACK, and some SET statements do not use a section.

segment. A group of pages that holds rows of a single table. See also *segmented table space*.

segmented table space. A table space that is divided into equal-sized groups of pages called segments. Segments are assigned to tables so that rows of different tables are never stored in the same segment.

self-referencing constraint. A referential constraint that defines a relationship in which a table is a dependent of itself.

self-referencing table. A table with a self-referencing constraint.

sensitive cursor. A cursor that is sensitive to changes that are made to the database after the result table has been materialized.

sequence. A user-defined object that generates a sequence of numeric values according to user specifications.

sequential data set. A non-DB2 data set whose records are organized on the basis of their successive physical positions, such as on magnetic tape. Several of the DB2 database utilities require sequential data sets.

sequential prefetch. A mechanism that triggers consecutive asynchronous I/O operations. Pages are fetched before they are required, and several pages are read with a single I/O operation.

serial cursor. A cursor that can be moved only in a forward direction.

serialized profile. A Java object that contains SQL statements and descriptions of host variables. The SQLJ translator produces a serialized profile for each connection context.

server. The target of a request from a remote requester. In the DB2 environment, the server function is provided by the distributed data facility, which is used to access DB2 data from remote applications.

server-side programming. A method for adding DB2 data into dynamic Web pages.

service class. An eight-character identifier that is used by the z/OS Workload Manager to associate user performance goals with a particular DDF thread or stored procedure. A service class is also used to classify work on parallelism assistants.

service request block. A unit of work that is scheduled to execute in another address space.

session. A link between two nodes in a VTAM network.

session protocols. The available set of SNA communication requests and responses.

shared communications area (SCA). A coupling facility list structure that a DB2 data sharing group uses for inter-DB2 communication.

share lock. A lock that prevents concurrently executing application processes from changing data, but not from reading data. Contrast with *exclusive lock*.

shift-in character. A special control character (X'0F') that is used in EBCDIC systems to denote that the subsequent bytes represent SBCS characters. See also *shift-out character*.

shift-out character. A special control character (X'0E') that is used in EBCDIC systems to denote that the subsequent bytes, up to the next shift-in control character, represent DBCS characters. See also *shift-in character*.

sign-on. A request that is made on behalf of an individual CICS or IMS application process by an attachment facility to enable DB2 to verify that it is authorized to use DB2 resources.

simple page set. A nonpartitioned page set. A simple page set initially consists of a single data set (page set piece). If and when that data set is extended to 2 GB, another data set is created, and so on, up to a total of 32 data sets. DB2 considers the data sets to be a single contiguous linear address space containing a maximum of 64 GB. Data is stored in the next available location within this address space without regard to any partitioning scheme.

simple table space. A table space that is neither partitioned nor segmented.

single-byte character set (SBCS). A set of characters in which each character is represented by a single byte. Contrast with *double-byte character set* or *multibyte character set*.

single-precision floating point number. A 32-bit approximate representation of a real number.

size. In the C language, the total number of digits in a decimal number (called the *precision* in SQL). The DB2 library uses the SQL term.

SMF. System Management Facilities.

SMP/E. System Modification Program/Extended.

SMS. Storage Management Subsystem.

SNA. Systems Network Architecture.

SNA network. The part of a network that conforms to the formats and protocols of Systems Network Architecture (SNA).

socket. A callable TCP/IP programming interface that TCP/IP network applications use to communicate with remote TCP/IP partners.

sourced function. A function that is implemented by another built-in or user-defined function that is already known to the database manager. This function can be a scalar function or a column (aggregating) function; it returns a single value from a set of values (for example, MAX or AVG). Contrast with *built-in function, external function,* and *SQL function*.

source program. A set of host language statements and SQL statements that is processed by an SQL precompiler.

source table. A table that can be a base table, a view, a table expression, or a user-defined table function.

source type. An existing type that DB2 uses to internally represent a distinct type.

space. A sequence of one or more blank characters.

special register. A storage area that DB2 defines for an application process to use for storing information that can be referenced in SQL statements. Examples of special registers are USER and CURRENT DATE.

specific function name. A particular user-defined function that is known to the database manager by its specific name. Many specific user-defined functions can have the same function name. When a user-defined function is defined to the database, every function is assigned a specific name that is unique within its schema. Either the user can provide this name, or a default name is used.

SPUFI. SQL Processor Using File Input.

SQL. Structured Query Language.

SQL authorization ID (SQL ID). The authorization ID that is used for checking dynamic SQL statements in some situations.

SQLCA. SQL communication area.

SQL communication area (SQLCA). A structure that is used to provide an application program with information about the execution of its SQL statements.

SQL connection. An association between an application process and a local or remote application server or database server.

SQLDA. SQL descriptor area.

SQL descriptor area (SQLDA). A structure that describes input variables, output variables, or the columns of a result table.

SQL escape character. The symbol that is used to enclose an SQL delimited identifier. This symbol is the double quotation mark ("). See also *escape character*.

SQL function. A user-defined function in which the CREATE FUNCTION statement contains the source code. The source code is a single SQL expression that evaluates to a single value. The SQL user-defined function can return only one parameter.

SQL ID. SQL authorization ID.

SQLJ. Structured Query Language (SQL) that is embedded in the Java programming language.

SQL path. An ordered list of schema names that are used in the resolution of unqualified references to user-defined functions, distinct types, and stored procedures. In dynamic SQL, the current path is found in the CURRENT PATH special register. In static SQL, it is defined in the PATH bind option.

SQL procedure. A user-written program that can be invoked with the SQL CALL statement. Contrast with *external procedure*.

SQL processing conversation. Any conversation that requires access of DB2 data, either through an application or by dynamic query requests.

SQL Processor Using File Input (SPUFI). A facility of the TSO attachment subcomponent that enables the DB2I user to execute SQL statements without embedding them in an application program.

SQL return code. Either SQLCODE or SQLSTATE.

SQL routine. A user-defined function or stored procedure that is based on code that is written in SQL.

SQL statement coprocessor. An alternative to the DB2 precompiler that lets the user process SQL statements at compile time. The user invokes an SQL statement coprocessor by specifying a compiler option.

SQL string delimiter. A symbol that is used to enclose an SQL string constant. The SQL string delimiter is the apostrophe ('), except in COBOL applications, where the user assigns the symbol, which is either an apostrophe or a double quotation mark (").

SRB. Service request block.

SSI. Subsystem interface (in z/OS).

SSM. Subsystem member (in IMS).

stand-alone. An attribute of a program that means that it is capable of executing separately from DB2, without using DB2 services.

star join. A method of joining a dimension column of a fact table to the key column of the corresponding dimension table. See also *join*, *dimension*, and *star schema*.

star schema. The combination of a fact table (which contains most of the data) and a number of dimension tables. See also *star join*, *dimension*, and *dimension table*.

statement handle. In DB2 ODBC, the data object that contains information about an SQL statement that is managed by DB2 ODBC. This includes information such as dynamic arguments, bindings for dynamic arguments and columns, cursor information, result values, and status information. Each statement handle is associated with the connection handle.

statement string. For a dynamic SQL statement, the character string form of the statement.

statement trigger. A trigger that is defined with the trigger granularity FOR EACH STATEMENT.

static cursor. A named control structure that does not change the size of the result table or the order of its rows after an application opens the cursor. Contrast with *dynamic cursor.*

static SQL. SQL statements, embedded within a program, that are prepared during the program preparation process (before the program is executed). After being prepared, the SQL statement does not change (although values of host variables that are specified by the statement might change).

storage group. A named set of disks on which DB2 data can be stored.

stored procedure. A user-written application program that can be invoked through the use of the SQL CALL statement.

string. See *character string* or *graphic string*.

strong typing. A process that guarantees that only user-defined functions and operations that are defined on a distinct type can be applied to that type. For example, you cannot directly compare two currency types, such as Canadian dollars and U.S. dollars. But you can provide a user-defined function to convert one currency to the other and then do the comparison.

structure. (1) A name that refers collectively to different types of DB2 objects, such as tables, databases, views, indexes, and table spaces. (2) A construct that uses z/OS to map and manage storage on a coupling facility. See also *cache structure*, *list structure*, or *lock structure*.

Structured Query Language (SQL). A standardized language for defining and manipulating data in a relational database.

structure owner. In relation to group buffer pools, the DB2 member that is responsible for the following activities:

- Coordinating rebuild, checkpoint, and damage assessment processing
- Monitoring the group buffer pool threshold and notifying castout owners when the threshold has been reached

subcomponent. A group of closely related DB2 modules that work together to provide a general function.

subject table. The table for which a trigger is created. When the defined triggering event occurs on this table, the trigger is activated.

subpage. The unit into which a physical index page can be divided.

subquery. A SELECT statement within the WHERE or HAVING clause of another SQL statement; a nested SQL statement.

subselect. That form of a query that does not include an ORDER BY clause, an UPDATE clause, or UNION operators.

substitution character. A unique character that is substituted during character conversion for any characters in the source program that do not have a match in the target coding representation.

subsystem. A distinct instance of a relational database management system (RDBMS).

surrogate pair. A coded representation for a single character that consists of a sequence of two 16-bit code units, in which the first value of the pair is a high-surrogate code unit in the range U+D800 through U+DBFF, and the second value is a low-surrogate code unit in the range U+DC00 through U+DFFF. Surrogate pairs provide an extension mechanism for encoding 917 476 characters without requiring the use of 32-bit characters.

SVC dump. A dump that is issued when a z/OS or a DB2 functional recovery routine detects an error.

sync point. See *commit point*.

syncpoint tree. The tree of recovery managers and resource managers that are involved in a logical unit of work, starting with the recovery manager, that make the final commit decision.

synonym. In SQL, an alternative name for a table or view. Synonyms can be used to refer only to objects at the subsystem in which the synonym is defined.

syntactic character set. A set of 81 graphic characters that are registered in the IBM registry as character set 00640. This set was originally recommended to the programming language community to be used for syntactic purposes toward maximizing portability and interchangeability across systems and country boundaries. It is contained in most of the primary registered character sets, with a few exceptions. See also *invariant character set*.

Sysplex. See *Parallel Sysplex*.

Sysplex query parallelism. Parallel execution of a single query that is accomplished by using multiple tasks on more than one DB2 subsystem. See also *query CP parallelism*.

system administrator. The person at a computer installation who designs, controls, and manages the use of the computer system.

system agent. A work request that DB2 creates internally such as prefetch processing, deferred writes, and service tasks.

system conversation. The conversation that two DB2 subsystems must establish to process system messages before any distributed processing can begin.

system diagnostic work area (SDWA). The data that is recorded in a SYS1.LOGREC entry that describes a program or hardware error.

system-directed connection. A connection that a relational DBMS manages by processing SQL statements with three-part names.

System Modification Program/Extended (SMP/E). A z/OS tool for making software changes in programming systems (such as DB2) and for controlling those changes.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information through and controlling the configuration and operation of networks.

SYS1.DUMPxx data set. A data set that contains a system dump (in z/OS).

SYS1.LOGREC. A service aid that contains important information about program and hardware errors (in z/OS).

T

table. A named data object consisting of a specific number of columns and some number of unordered rows. See also *base table* or *temporary table*.

table-controlled partitioning. A type of partitioning in which partition boundaries for a partitioned table are controlled by values that are defined in the CREATE TABLE statement. Partition limits are saved in the LIMITKEY_INTERNAL column of the SYSIBM.SYSTABLEPART catalog table.

table function. A function that receives a set of arguments and returns a table to the SQL statement that references the function. A table function can be referenced only in the FROM clause of a subselect.

table locator. A mechanism that allows access to trigger transition tables in the FROM clause of SELECT statements, in the subselect of INSERT statements, or from within user-defined functions. A table locator is a fullword integer value that represents a transition table.

table space. A page set that is used to store the records in one or more tables.

table space set. A set of table spaces and partitions that should be recovered together for one of these reasons:

- Each of them contains a table that is a parent or descendent of a table in one of the others.
- The set contains a base table and associated auxiliary tables.

A table space set can contain both types of relationships.

task control block (TCB). A z/OS control block that is used to communicate information about tasks within an address space that are connected to DB2. See also *address space connection*.

TB. Terabyte (1 099 511 627 776 bytes).

TCB. Task control block (in z/OS).

TCP/IP. A network communication protocol that computer systems use to exchange information across telecommunication links.

TCP/IP port. A 2-byte value that identifies an end user or a TCP/IP network application within a TCP/IP host.

template. A DB2 utilities output data set descriptor that is used for dynamic allocation. A template is defined by the TEMPLATE utility control statement.

temporary table. A table that holds temporary data. Temporary tables are useful for holding or sorting intermediate results from queries that contain a large number of rows. The two types of temporary table, which are created by different SQL statements, are the created temporary table and the declared temporary table. Contrast with *result table*. See also *created temporary table* and *declared temporary table*.

Terminal Monitor Program (TMP). A program that provides an interface between terminal users and command processors and has access to many system services (in z/OS).

thread. The DB2 structure that describes an application's connection, traces its progress, processes resource functions, and delimits its accessibility to DB2 resources and services. Most DB2 functions execute under a thread structure. See also *allied thread* and *database access thread*.

threadsafe. A characteristic of code that allows multithreading both by providing private storage areas for each thread, and by properly serializing shared (global) storage areas.

three-part name. The full name of a table, view, or alias. It consists of a location name, authorization ID, and an object name, separated by a period.

time. A three-part value that designates a time of day in hours, minutes, and seconds.

time duration. A decimal integer that represents a number of hours, minutes, and seconds.

timeout. Abnormal termination of either the DB2 subsystem or of an application because of the unavailability of resources. Installation specifications are set to determine both the amount of time DB2 is to wait for IRLM services after starting, and the amount of time IRLM is to wait if a resource that an application requests is unavailable. If either of these time specifications is exceeded, a timeout is declared.

Time-Sharing Option (TSO). An option in MVS that provides interactive time sharing from remote terminals.

timestamp. A seven-part value that consists of a date and time. The timestamp is expressed in years, months, days, hours, minutes, seconds, and microseconds.

TMP. Terminal Monitor Program.

to-do. A state of a unit of recovery that indicates that the unit of recovery's changes to recoverable DB2 resources are indoubt and must either be applied to the disk media or backed out, as determined by the commit coordinator.

trace. A DB2 facility that provides the ability to monitor and collect DB2 monitoring, auditing, performance, accounting, statistics, and serviceability (global) data.

transaction lock. A lock that is used to control concurrent execution of SQL statements.

transaction program name. In SNA LU 6.2 conversations, the name of the program at the remote logical unit that is to be the other half of the conversation.

transient XML data type. A data type for XML values that exists only during query processing.

transition table. A temporary table that contains all the affected rows of the subject table in their state before or after the triggering event occurs. Triggered SQL statements in the trigger definition can reference the table of changed rows in the old state or the new state.

transition variable. A variable that contains a column value of the affected row of the subject table in its state before or after the triggering event occurs. Triggered SQL statements in the trigger definition can reference the set of old values or the set of new values.

tree structure. A data structure that represents entities in nodes, with a most one parent node for each node, and with only one root node.

trigger. A set of SQL statements that are stored in a DB2 database and executed when a certain event occurs in a DB2 table.

trigger activation. The process that occurs when the trigger event that is defined in a trigger definition is executed. Trigger activation consists of the evaluation of the triggered action condition and conditional execution of the triggered SQL statements.

trigger activation time. An indication in the trigger definition of whether the trigger should be activated before or after the triggered event.

trigger body. The set of SQL statements that is executed when a trigger is activated and its triggered action condition evaluates to true. A trigger body is also called *triggered SQL statements*.

trigger cascading. The process that occurs when the triggered action of a trigger causes the activation of another trigger.

triggered action. The SQL logic that is performed when a trigger is activated. The triggered action consists of an optional triggered action condition and a set of triggered SQL statements that are executed only if the condition evaluates to true.

triggered action condition. An optional part of the triggered action. This Boolean condition appears as a WHEN clause and specifies a condition that DB2 evaluates to determine if the triggered SQL statements should be executed.

triggered SQL statements. The set of SQL statements that is executed when a trigger is activated and its triggered action condition evaluates to true. Triggered SQL statements are also called the *trigger body*.

trigger granularity. A characteristic of a trigger, which determines whether the trigger is activated:
- Only once for the triggering SQL statement
- Once for each row that the SQL statement modifies

triggering event. The specified operation in a trigger definition that causes the activation of that trigger. The triggering event is comprised of a triggering operation (INSERT, UPDATE, or DELETE) and a subject table on which the operation is performed.

triggering SQL operation. The SQL operation that causes a trigger to be activated when performed on the subject table.

trigger package. A package that is created when a CREATE TRIGGER statement is executed. The package is executed when the trigger is activated.

TSO. Time-Sharing Option.

TSO attachment facility. A DB2 facility consisting of the DSN command processor and DB2I. Applications that are not written for the CICS or IMS environments can run under the TSO attachment facility.

typed parameter marker. A parameter marker that is specified along with its target data type. It has the general form:

```
CAST(? AS data-type)
```

type 1 indexes. Indexes that were created by a release of DB2 before DB2 Version 4 or that are specified as type 1 indexes in Version 4. Contrast with *type 2 indexes*. As of Version 8, type 1 indexes are no longer supported.

type 2 indexes. Indexes that are created on a release of DB2 after Version 7 or that are specified as type 2 indexes in Version 4 or later.

U

UCS-2. Universal Character Set, coded in 2 octets, which means that characters are represented in 16-bits per character.

UDF. User-defined function.

UDT. User-defined data type. In DB2 UDB for z/OS, the term *distinct type* is used instead of user-defined data type. See *distinct type*.

uncommitted read (UR). The isolation level that allows an application to read uncommitted data.

underlying view. The view on which another view is directly or indirectly defined.

undo. A state of a unit of recovery that indicates that the changes that the unit of recovery made to recoverable DB2 resources must be backed out.

Unicode. A standard that parallels the ISO-10646 standard. Several implementations of the Unicode standard exist, all of which have the ability to represent a large percentage of the characters that are contained in the many scripts that are used throughout the world.

uniform resource locator (URL). A Web address, which offers a way of naming and locating specific items on the Web.

union. An SQL operation that combines the results of two SELECT statements. Unions are often used to merge lists of values that are obtained from several tables.

unique constraint. An SQL rule that no two values in a primary key, or in the key of a unique index, can be the same.

unique index. An index that ensures that no identical key values are stored in a column or a set of columns in a table.

unit of recovery. A recoverable sequence of operations within a single resource manager, such as an instance of DB2. Contrast with *unit of work*.

unit of recovery identifier (URID). The LOGRBA of the first log record for a unit of recovery. The URID also appears in all subsequent log records for that unit of recovery.

unit of work. A recoverable sequence of operations within an application process. At any time, an application process is a single unit of work, but the life of an application process can involve many units of work as a result of commit or rollback operations. In a *multisite update* operation, a single unit of work can include several *units of recovery*. Contrast with *unit of recovery*.

Universal Unique Identifier (UUID). An identifier that is immutable and unique across time and space (in z/OS).

unlock. The act of releasing an object or system resource that was previously locked and returning it to general availability within DB2.

untyped parameter marker. A parameter marker that is specified without its target data type. It has the form of a single question mark (?).

updatability. The ability of a cursor to perform positioned updates and deletes. The updatability of a cursor can be influenced by the SELECT statement and the cursor sensitivity option that is specified on the DECLARE CURSOR statement.

update hole. The location on which a cursor is positioned when a row in a result table is fetched again and the new values no longer satisfy the search condition. DB2 marks a row in the result table as an update hole when an update to the corresponding row in the database causes that row to no longer qualify for the result table.

update trigger. A trigger that is defined with the triggering SQL operation UPDATE.

upstream. The node in the syncpoint tree that is responsible, in addition to other recovery or resource managers, for coordinating the execution of a two-phase commit.

UR. Uncommitted read.

URE. Unit of recovery element.

URID . Unit of recovery identifier.

URL. Uniform resource locator.

user-defined data type (UDT). See *distinct type*.

user-defined function (UDF). A function that is defined to DB2 by using the CREATE FUNCTION statement and that can be referenced thereafter in SQL statements. A user-defined function can be an *external function,* a *sourced function,* or an *SQL function.* Contrast with *built-in function*.

user view. In logical data modeling, a model or representation of critical information that the business requires.

UTF-8. Unicode Transformation Format, 8-bit encoding form, which is designed for ease of use with existing ASCII-based systems. The CCSID value for data in UTF-8 format is 1208. DB2 UDB for z/OS supports UTF-8 in mixed data fields.

UTF-16. Unicode Transformation Format, 16-bit encoding form, which is designed to provide code values for over a million characters and a superset of UCS-2. The CCSID value for data in UTF-16 format is 1200. DB2 UDB for z/OS supports UTF-16 in graphic data fields.

UUID. Universal Unique Identifier.

V

value. The smallest unit of data that is manipulated in SQL.

variable. A data element that specifies a value that can be changed. A COBOL elementary data item is an example of a variable. Contrast with *constant*.

variant function. See *nondeterministic function*.

varying-length string. A character or graphic string whose length varies within set limits. Contrast with *fixed-length string*.

version. A member of a set of similar programs, DBRMs, packages, or LOBs.

A version of a program is the source code that is produced by precompiling the program. The program version is identified by the program name and a timestamp (consistency token).

A version of a DBRM is the DBRM that is produced by precompiling a program. The DBRM version is identified by the same program name and timestamp as a corresponding program version.

A version of a package is the result of binding a DBRM within a particular database system. The package version is identified by the same program name and consistency token as the DBRM.

A version of a LOB is a copy of a LOB value at a point in time. The version number for a LOB is stored in the auxiliary index entry for the LOB.

view. An alternative representation of data from one or more tables. A view can include all or some of the columns that are contained in tables on which it is defined.

view check option. An option that specifies whether every row that is inserted or updated through a view must conform to the definition of that view. A view check option can be specified with the WITH CASCADED CHECK OPTION, WITH CHECK OPTION, or WITH LOCAL CHECK OPTION clauses of the CREATE VIEW statement.

Virtual Storage Access Method (VSAM). An access method for direct or sequential processing of fixed- and varying-length records on disk devices. The records in a VSAM data set or file can be organized in logical sequence by a key field (key sequence), in the physical sequence in which they are written on the data set or file (entry-sequence), or by relative-record number (in z/OS).

Virtual Telecommunications Access Method (VTAM). An IBM licensed program that controls communication and the flow of data in an SNA network (in z/OS).

| **volatile table.** A table for which SQL operations
| choose index access whenever possible.

VSAM. Virtual Storage Access Method.

VTAM. Virtual Telecommunication Access Method (in z/OS).

W

warm start. The normal DB2 restart process, which involves reading and processing log records so that data that is under the control of DB2 is consistent. Contrast with *cold start*.

WLM application environment. A z/OS Workload Manager attribute that is associated with one or more stored procedures. The WLM application environment determines the address space in which a given DB2 stored procedure runs.

write to operator (WTO). An optional user-coded service that allows a message to be written to the system console operator informing the operator of errors and unusual system conditions that might need to be corrected (in z/OS).

WTO. Write to operator.

WTOR. Write to operator (WTO) with reply.

X

XCF. See *cross-system coupling facility*.

XES. See *cross-system extended services*.

| **XML.** See *Extensible Markup Language*.

| **XML attribute.** A name-value pair within a tagged
| XML element that modifies certain features of the
| element.

XML element. A logical structure in an XML
document that is delimited by a start and an end tag.
Anything between the start tag and the end tag is the
content of the element.

XML node. The smallest unit of valid, complete structure in a document. For example, a node can represent an element, an attribute, or a text string.

| **XML publishing functions.** Functions that return
| XML values from SQL values.

X/Open. An independent, worldwide open systems organization that is supported by most of the world's largest information systems suppliers, user organizations, and software companies. X/Open's goal is to increase the portability of applications by combining existing and emerging standards.

XRF. Extended recovery facility.

Z

| **z/OS.** An operating system for the eServer™ product
| line that supports 64-bit real and virtual storage.

z/OS Distributed Computing Environment (z/OS DCE). A set of technologies that are provided by the Open Software Foundation to implement distributed computing.

Bibliography

DB2 Universal Database for z/OS Version 8 product information:

- *DB2 Administration Guide*, SC18-7413
- *DB2 Application Programming and SQL Guide*, SC18-7415
- *DB2 Application Programming Guide and Reference for Java*, SC18-7414
- *DB2 Codes*, GC18-9603
- *DB2 Command Reference*, SC18-7416
- *DB2 Common Criteria Guide*, SC18-9672
- *DB2 Data Sharing: Planning and Administration*, SC18-7417
- *DB2 Diagnosis Guide and Reference*, LY37-3201
- *DB2 Diagnostic Quick Reference Card*, LY37-3202
- *DB2 Image, Audio, and Video Extenders Administration and Programming*, SC26-9947
- *DB2 Installation Guide*, GC18-7418
- *DB2 Licensed Program Specifications*, GC18-7420
- *DB2 Management Clients Package Program Directory*, GI10-8567
- *DB2 Messages*, GC18-9602
- *DB2 ODBC Guide and Reference*, SC18-7423
- *The Official Introduction to DB2 UDB for z/OS*
- *DB2 Program Directory*, GI10-8566
- *DB2 RACF Access Control Module Guide*, SC18-7433
- *DB2 Reference for Remote DRDA Requesters and Servers*, SC18-7424
- *DB2 Reference Summary*, SX26-3853
- *DB2 Release Planning Guide*, SC18-7425
- *DB2 SQL Reference*, SC18-7426
- *DB2 Text Extender Administration and Programming*, SC26-9948
- *DB2 Utility Guide and Reference*, SC18-7427
- *DB2 What's New?*, GC18-7428
- *DB2 XML Extender for z/OS Administration and Programming*, SC18-7431

Books and resources about related products:

APL2®

- *APL2 Programming Guide*, SH21-1072
- *APL2 Programming: Language Reference*, SH21-1061
- *APL2 Programming: Using Structured Query Language (SQL)*, SH21-1057

BookManager® READ/MVS

- *BookManager READ/MVS V1R3: Installation Planning & Customization*, SC38-2035

C language: IBM C/C++ for z/OS

- *z/OS C/C++ Programming Guide*, SC09-4765
- *z/OS C/C++ Run-Time Library Reference*, SA22-7821

Character Data Representation Architecture

- *Character Data Representation Architecture Overview*, GC09-2207
- *Character Data Representation Architecture Reference and Registry*, SC09-2190

CICS Transaction Server for z/OS

The publication order numbers below are for Version 2 Release 2 and Version 2 Release 3 (with the release 2 number listed first).

- *CICS Transaction Server for z/OS Information Center*, SK3T-6903 or SK3T-6957.
- *CICS Transaction Server for z/OS Application Programming Guide*, SC34-5993 or SC34-6231
- *CICS Transaction Server for z/OS Application Programming Reference*, SC34-5994 or SC34-6232
- *CICS Transaction Server for z/OS CICS-RACF Security Guide*, SC34-6011 or SC34-6249
- *CICS Transaction Server for z/OS CICS Supplied Transactions*, SC34-5992 or SC34-6230
- *CICS Transaction Server for z/OS Customization Guide*, SC34-5989 or SC34-6227
- *CICS Transaction Server for z/OS Data Areas*, LY33-6100 or LY33-6103
- *CICS Transaction Server for z/OS DB2 Guide*, SC34-6014 or SC34-6252
- *CICS Transaction Server for z/OS External Interfaces Guide*, SC34-6006 or SC34-6244
- *CICS Transaction Server for z/OS Installation Guide*, GC34-5985 or GC34-6224
- *CICS Transaction Server for z/OS Intercommunication Guide*, SC34-6005 or SC34-6243
- *CICS Transaction Server for z/OS Messages and Codes*, GC34-6003 or GC34-6241
- *CICS Transaction Server for z/OS Operations and Utilities Guide*, SC34-5991 or SC34-6229

© Copyright IBM Corp. 1983, 2008

- *CICS Transaction Server for z/OS Performance Guide*, SC34-6009 or SC34-6247
- *CICS Transaction Server for z/OS Problem Determination Guide*, SC34-6002 or SC34-6239
- *CICS Transaction Server for z/OS Release Guide*, GC34-5983 or GC34-6218
- *CICS Transaction Server for z/OS Resource Definition Guide*, SC34-5990 or SC34-6228
- *CICS Transaction Server for z/OS System Definition Guide*, SC34-5988 or SC34–6226
- *CICS Transaction Server for z/OS System Programming Reference*, SC34-5595 or SC34–6233

CICS Transaction Server for OS/390
- *CICS Transaction Server for OS/390 Application Programming Guide*, SC33-1687
- *CICS Transaction Server for OS/390 DB2 Guide*, SC33-1939
- *CICS Transaction Server for OS/390 External Interfaces Guide*, SC33-1944
- *CICS Transaction Server for OS/390 Resource Definition Guide*, SC33-1684

COBOL:
- *IBM COBOL Language Reference*, SC27-1408
- *Enterprise COBOL for z/OS Programming Guide*, SC27-1412

Database Design
- *DB2 for z/OS and OS/390 Development for Performance Volume I* by Gabrielle Wiorkowski, Gabrielle & Associates, ISBN 0-96684-605-2
- *DB2 for z/OS and OS/390 Development for Performance Volume II* by Gabrielle Wiorkowski, Gabrielle & Associates, ISBN 0-96684-606-0
- *Handbook of Relational Database Design* by C. Fleming and B. Von Halle, Addison Wesley, ISBN 0-20111-434-8

DB2 Administration Tool
- *DB2 Administration Tool for z/OS User's Guide and Reference*, available on the Web at www.ibm.com/software/data/db2imstools/library.html

DB2 Buffer Pool Analyzer for z/OS
- *DB2 Buffer Pool Tool for z/OS User's Guide and Reference*, available on the Web at www.ibm.com/software/data/db2imstools/library.html

DB2 Connect™
- *IBM DB2 Connect Quick Beginnings for DB2 Connect Enterprise Edition*, GC09-4833
- *IBM DB2 Connect Quick Beginnings for DB2 Connect Personal Edition*, GC09-4834
- *IBM DB2 Connect User's Guide*, SC09-4835

DB2 DataPropagator™
- *DB2 Universal Database Replication Guide and Reference*, SC27-1121

DB2 Performance Expert for z/OS, Version 1

The following books are part of the DB2 Performance Expert library. Some of these books include information about the following tools: IBM DB2 Performance Expert for z/OS; IBM DB2 Performance Monitor for z/OS; and DB2 Buffer Pool Analyzer for z/OS.
- *OMEGAMON Buffer Pool Analyzer User's Guide*, SC18-7972
- *OMEGAMON Configuration and Customization*, SC18-7973
- *OMEGAMON Messages*, SC18-7974
- *OMEGAMON Monitoring Performance from ISPF*, SC18-7975
- *OMEGAMON Monitoring Performance from Performance Expert Client*, SC18-7976
- *OMEGAMON Program Directory*, GI10-8549
- *OMEGAMON Report Command Reference*, SC18-7977
- *OMEGAMON Report Reference*, SC18-7978
- *Using IBM Tivoli OMEGAMON XE on z/OS*, SC18-7979

DB2 Query Management Facility (QMF) Version 8.1
- *DB2 Query Management Facility: DB2 QMF High Performance Option User's Guide for TSO/CICS*, SC18-7450
- *DB2 Query Management Facility: DB2 QMF Messages and Codes*, GC18-7447
- *DB2 Query Management Facility: DB2 QMF Reference*, SC18-7446
- *DB2 Query Management Facility: Developing DB2 QMF Applications*, SC18-7651
- *DB2 Query Management Facility: Getting Started with DB2 QMF for Windows and DB2 QMF for WebSphere*, SC18-7449
- *DB2 Query Management Facility: Getting Started with DB2 QMF Query Miner*, GC18-7451
- *DB2 Query Management Facility: Installing and Managing DB2 QMF for TSO/CICS*, GC18-7444
- *DB2 Query Management Facility: Installing and Managing DB2 QMF for Windows and DB2 QMF for WebSphere*, GC18-7448

- *DB2 Query Management Facility: Introducing DB2 QMF*, GC18-7443
- *DB2 Query Management Facility: Using DB2 QMF*, SC18-7445
- *DB2 Query Management Facility: DB2 QMF Visionary Developer's Guide*, SC18-9093
- *DB2 Query Management Facility: DB2 QMF Visionary Getting Started Guide*, GC18-9092

DB2 Redbooks™

For access to all IBM Redbooks about DB2, see the IBM Redbooks Web page at www.ibm.com/redbooks

DB2 Server for VSE & VM
- *DB2 Server for VM: DBS Utility*, SC09-2983

DB2 Universal Database Cross-Platform information
- *IBM DB2 Universal Database SQL Reference for Cross-Platform Development*, available at www.ibm.com/software/data/developer/cpsqlref/

DB2 Universal Database for iSeries

The following books are available at www.ibm.com/iseries/infocenter
- *DB2 Universal Database for iSeries Performance and Query Optimization*
- *DB2 Universal Database for iSeries Database Programming*
- *DB2 Universal Database for iSeries SQL Programming Concepts*
- *DB2 Universal Database for iSeries SQL Programming with Host Languages*
- *DB2 Universal Database for iSeries SQL Reference*
- *DB2 Universal Database for iSeries Distributed Data Management*
- *DB2 Universal Database for iSeries Distributed Database Programming*

DB2 Universal Database for Linux, UNIX, and Windows:
- *DB2 Universal Database Administration Guide: Planning*, SC09-4822
- *DB2 Universal Database Administration Guide: Implementation*, SC09-4820
- *DB2 Universal Database Administration Guide: Performance*, SC09-4821
- *DB2 Universal Database Administrative API Reference*, SC09-4824
- *DB2 Universal Database Application Development Guide: Building and Running Applications*, SC09-4825
- *DB2 Universal Database Call Level Interface Guide and Reference, Volumes 1 and 2*, SC09-4849 and SC09-4850
- *DB2 Universal Database Command Reference*, SC09-4828
- *DB2 Universal Database SQL Reference Volume 1*, SC09-4844
- *DB2 Universal Database SQL Reference Volume 2*, SC09-4845

Device Support Facilities
- *Device Support Facilities User's Guide and Reference*, GC35-0033

DFSMS

These books provide information about a variety of components of DFSMS, including z/OS DFSMS, z/OS DFSMSdfp™, z/OS DFSMSdss, z/OS DFSMShsm, and z/OS DFP.
- *z/OS DFSMS Access Method Services for Catalogs*, SC26-7394
- *z/OS DFSMSdss Storage Administration Guide*, SC35-0423
- *z/OS DFSMSdss Storage Administration Reference*, SC35-0424
- *z/OS DFSMShsm Managing Your Own Data*, SC35-0420
- *z/OS DFSMSdfp: Using DFSMSdfp in the z/OS Environment*, SC26-7473
- *z/OS DFSMSdfp Diagnosis Reference*, GY27-7618
- *z/OS DFSMS: Implementing System-Managed Storage*, SC27-7407
- *z/OS DFSMS: Macro Instructions for Data Sets*, SC26-7408
- *z/OS DFSMS: Managing Catalogs*, SC26-7409
- *z/OS MVS: Program Management User's Guide and Reference*, SA22-7643
- *z/OS MVS Program Management: Advanced Facilities*, SA22-7644
- *z/OS DFSMSdfp Storage Administration Reference*, SC26-7402
- *z/OS DFSMS: Using Data Sets*, SC26-7410
- *DFSMSdfp Advanced Services* , SC26-7400
- *DFSMS/MVS: Utilities*, SC26-7414

DFSORT™
- *DFSORT Application Programming: Guide*, SC33-4035
- *DFSORT Installation and Customization*, SC33-4034

Distributed Relational Database Architecture

- *Open Group Technical Standard*; the Open Group presently makes the following DRDA books available through its Web site at www.opengroup.org
  - *Open Group Technical Standard, DRDA Version 3 Vol. 1: Distributed Relational Database Architecture*
  - *Open Group Technical Standard, DRDA Version 3 Vol. 2: Formatted Data Object Content Architecture*
  - *Open Group Technical Standard, DRDA Version 3 Vol. 3: Distributed Data Management Architecture*

Domain Name System
- *DNS and BIND, Third Edition, Paul Albitz and Cricket Liu, O'Reilly*, ISBN 0-59600-158-4

Education
- Information about IBM educational offerings is available on the Web at http://www.ibm.com/software/sw-training/
- A collection of glossaries of IBM terms is available on the IBM Terminology Web site at www.ibm.com/ibm/terminology/index.html

eServer zSeries®
- *IBM eServer zSeries Processor Resource/System Manager Planning Guide*, SB10-7033

Fortran: VS Fortran
- *VS Fortran Version 2: Language and Library Reference*, SC26-4221
- *VS Fortran Version 2: Programming Guide for CMS and MVS*, SC26-4222

High Level Assembler
- *High Level Assembler for MVS and VM and VSE Language Reference*, SC26-4940
- *High Level Assembler for MVS and VM and VSE Programmer's Guide*, SC26-4941

ICSF
- *z/OS ICSF Overview*, SA22-7519
- *Integrated Cryptographic Service Facility Administrator's Guide*, SA22-7521

IMS Version 8

IMS product information is available on the IMS Library Web page, which you can find at www.ibm.com/ims
- *IMS Administration Guide: System*, SC27-1284
- *IMS Administration Guide: Transaction Manager*, SC27-1285
- *IMS Application Programming: Database Manager*, SC27-1286
- *IMS Application Programming: Design Guide*, SC27-1287
- *IMS Application Programming: Transaction Manager*, SC27-1289
- *IMS Command Reference*, SC27-1291
- *IMS Customization Guide*, SC27-1294
- *IMS Install Volume 1: Installation Verification*, GC27-1297
- *IMS Install Volume 2: System Definition and Tailoring*, GC27-1298
- *IMS Messages and Codes Volumes 1 and 2*, GC27-1301 and GC27-1302
- *IMS Open Transaction Manager Access Guide and Reference*, SC18-7829
- *IMS Utilities Reference: System*, SC27-1309

General information about IMS Batch Terminal Simulator for z/OS is available on the Web at www.ibm.com/software/data/db2imstools/library.html

IMS DataPropagator
- *IMS DataPropagator for z/OS Administrator's Guide for Log*, SC27-1216
- *IMS DataPropagator: An Introduction*, GC27-1211
- *IMS DataPropagator for z/OS Reference*, SC27-1210

ISPF
- *z/OS ISPF Dialog Developer's Guide*, SC23-4821
- *z/OS ISPF Messages and Codes*, SC34-4815
- *z/OS ISPF Planning and Customizing*, GC34-4814
- *z/OS ISPF User's Guide Volumes 1 and 2*, SC34-4822 and SC34-4823

Language Environment
- *Debug Tool User's Guide and Reference*, SC18-7171
- *Debug Tool for z/OS and OS/390 Reference and Messages*, SC18-7172
- *z/OS Language Environment Concepts Guide*, SA22-7567
- *z/OS Language Environment Customization*, SA22-7564
- *z/OS Language Environment Debugging Guide*, GA22-7560
- *z/OS Language Environment Programming Guide*, SA22-7561
- *z/OS Language Environment Programming Reference*, SA22-7562

MQSeries®
- *MQSeries Application Messaging Interface*, SC34-5604

- *MQSeries for OS/390 Concepts and Planning Guide*, GC34-5650
- *MQSeries for OS/390 System Setup Guide*, SC34-5651

National Language Support
- *National Language Design Guide Volume 1*, SE09-8001
- *IBM National Language Support Reference Manual Volume 2*, SE09-8002

NetView®
- *Tivoli NetView for z/OS Installation: Getting Started*, SC31-8872
- *Tivoli NetView for z/OS User's Guide*, GC31-8849

Microsoft ODBC

Information about Microsoft ODBC is available at http://msdn.microsoft.com/library/

Parallel Sysplex Library
- *System/390 9672 Parallel Transaction Server, 9672 Parallel Enterprise Server, 9674 Coupling Facility System Overview For R1/R2/R3 Based Models*, SB10-7033
- *z/OS Parallel Sysplex Application Migration*, SA22-7662
- *z/OS Parallel Sysplex Overview: An Introduction to Data Sharing and Parallelism*, SA22-7661
- *z/OS Parallel Sysplex Test Report*, SA22-7663

The *Parallel Sysplex Configuration Assistant* is available at www.ibm.com/s390/pso/psotool

PL/I: Enterprise PL/I for z/OS
- *IBM Enterprise PL/I for z/OS Language Reference*, SC27-1460
- *IBM Enterprise PL/I for z/OS Programming Guide*, SC27-1457

PL/I: PL/I for MVS & VM
- *PL/I for MVS & VM Programming Guide*, SC26-3113

SMP/E
- *SMP/E for z/OS and OS/390 Reference*, SA22-7772
- *SMP/E for z/OS and OS/390 User's Guide*, SA22-7773

Storage Management
- *z/OS DFSMS: Implementing System-Managed Storage*, SC26-7407
- *MVS/ESA Storage Management Library: Managing Data*, SC26-7397
- *MVS/ESA Storage Management Library: Managing Storage Groups*, SC35-0421
- *MVS Storage Management Library: Storage Management Subsystem Migration Planning Guide*, GC26-7398

System Network Architecture (SNA)
- *SNA Formats*, GA27-3136
- *SNA LU 6.2 Peer Protocols Reference*, SC31-6808
- *SNA Transaction Programmer's Reference Manual for LU Type 6.2*, GC30-3084
- *SNA/Management Services Alert Implementation Guide*, GC31-6809

TCP/IP
- *IBM TCP/IP for MVS: Customization & Administration Guide*, SC31-7134
- *IBM TCP/IP for MVS: Diagnosis Guide*, LY43-0105
- *IBM TCP/IP for MVS: Messages and Codes*, SC31-7132
- *IBM TCP/IP for MVS: Planning and Migration Guide*, SC31-7189

TotalStorage™ Enterprise Storage Server
- *RAMAC Virtual Array: Implementing Peer-to-Peer Remote Copy*, SG24-5680
- *Enterprise Storage Server Introduction and Planning*, GC26-7444
- *IBM RAMAC Virtual Array*, SG24-6424

Unicode
- *z/OS Support for Unicode: Using Conversion Services*, SA22-7649

Information about Unicode, the Unicode consortium, the Unicode standard, and standards conformance requirements is available at www.unicode.org

VTAM
- *Planning for NetView, NCP, and VTAM*, SC31-8063
- *VTAM for MVS/ESA Diagnosis*, LY43-0078
- *VTAM for MVS/ESA Messages and Codes*, GC31-8369
- *VTAM for MVS/ESA Network Implementation Guide*, SC31-8370
- *VTAM for MVS/ESA Operation*, SC31-8372
- *z/OS Communications Server SNA Programming*, SC31-8829
- *z/OS Communicatons Server SNA Programmer's LU 6.2 Reference*, SC31-8810
- *VTAM for MVS/ESA Resource Definition Reference*, SC31-8377

WebSphere® family

- *WebSphere MQ Integrator Broker: Administration Guide*, SC34-6171
- *WebSphere MQ Integrator Broker for z/OS: Customization and Administration Guide*, SC34-6175
- *WebSphere MQ Integrator Broker: Introduction and Planning*, GC34-5599
- *WebSphere MQ Integrator Broker: Using the Control Center*, SC34-6168

z/Architecture™

- *z/Architecture Principles of Operation*, SA22-7832

z/OS

- *z/OS C/C++ Programming Guide*, SC09-4765
- *z/OS C/C++ Run-Time Library Reference*, SA22-7821
- *z/OS C/C++ User's Guide*, SC09-4767
- *z/OS Communications Server: IP Configuration Guide*, SC31-8875
- *z/OS DCE Administration Guide*, SC24-5904
- *z/OS DCE Introduction*, GC24-5911
- *z/OS DCE Messages and Codes*, SC24-5912
- *z/OS Information Roadmap*, SA22-7500
- *z/OS Introduction and Release Guide*, GA22-7502
- *z/OS JES2 Initialization and Tuning Guide*, SA22-7532
- *z/OS JES3 Initialization and Tuning Guide*, SA22-7549
- *z/OS Language Environment Concepts Guide*, SA22-7567
- *z/OS Language Environment Customization*, SA22-7564
- *z/OS Language Environment Debugging Guide*, GA22-7560
- *z/OS Language Environment Programming Guide*, SA22-7561
- *z/OS Language Environment Programming Reference*, SA22-7562
- *z/OS Managed System Infrastructure for Setup User's Guide*, SC33-7985
- *z/OS MVS Diagnosis: Procedures*, GA22-7587
- *z/OS MVS Diagnosis: Reference*, GA22-7588
- *z/OS MVS Diagnosis: Tools and Service Aids*, GA22-7589
- *z/OS MVS Initialization and Tuning Guide*, SA22-7591
- *z/OS MVS Initialization and Tuning Reference*, SA22-7592
- *z/OS MVS Installation Exits*, SA22-7593
- *z/OS MVS JCL Reference*, SA22-7597
- *z/OS MVS JCL User's Guide*, SA22-7598
- *z/OS MVS Planning: Global Resource Serialization*, SA22-7600
- *z/OS MVS Planning: Operations*, SA22-7601
- *z/OS MVS Planning: Workload Management*, SA22-7602
- *z/OS MVS Programming: Assembler Services Guide*, SA22-7605
- *z/OS MVS Programming: Assembler Services Reference, Volumes 1 and 2*, SA22-7606 and SA22-7607
- *z/OS MVS Programming: Authorized Assembler Services Guide*, SA22-7608
- *z/OS MVS Programming: Authorized Assembler Services Reference Volumes 1-4*, SA22-7609, SA22-7610, SA22-7611, and SA22-7612
- *z/OS MVS Programming: Callable Services for High-Level Languages*, SA22-7613
- *z/OS MVS Programming: Extended Addressability Guide*, SA22-7614
- *z/OS MVS Programming: Sysplex Services Guide*, SA22-7617
- *z/OS MVS Programming: Sysplex Services Reference*, SA22-7618
- *z/OS MVS Programming: Workload Management Services*, SA22-7619
- *z/OS MVS Recovery and Reconfiguration Guide*, SA22-7623
- *z/OS MVS Routing and Descriptor Codes*, SA22-7624
- *z/OS MVS Setting Up a Sysplex*, SA22-7625
- *z/OS MVS System Codes* SA22-7626
- *z/OS MVS System Commands*, SA22-7627
- *z/OS MVS System Messages Volumes 1-10*, SA22-7631, SA22-7632, SA22-7633, SA22-7634, SA22-7635, SA22-7636, SA22-7637, SA22-7638, SA22-7639, and SA22-7640
- *z/OS MVS Using the Subsystem Interface*, SA22-7642
- *z/OS Planning for Multilevel Security and the Common Criteria*, SA22-7509
- *z/OS RMF User's Guide*, SC33-7990
- *z/OS Security Server Network Authentication Server Administration*, SC24-5926
- *z/OS Security Server RACF Auditor's Guide*, SA22-7684
- *z/OS Security Server RACF Command Language Reference*, SA22-7687
- *z/OS Security Server RACF Macros and Interfaces*, SA22-7682
- *z/OS Security Server RACF Security Administrator's Guide*, SA22-7683
- *z/OS Security Server RACF System Programmer's Guide*, SA22-7681
- *z/OS Security Server RACROUTE Macro Reference*, SA22-7692
- *z/OS Support for Unicode: Using Conversion Services*, SA22-7649
- *z/OS TSO/E CLISTs*, SA22-7781
- *z/OS TSO/E Command Reference*, SA22-7782

- *z/OS TSO/E Customization*, SA22-7783
- *z/OS TSO/E Messages*, SA22-7786
- *z/OS TSO/E Programming Guide*, SA22-7788
- *z/OS TSO/E Programming Services*, SA22-7789
- *z/OS TSO/E REXX Reference*, SA22-7790
- *z/OS TSO/E User's Guide*, SA22-7794
- *z/OS UNIX System Services Command Reference*, SA22-7802
- *z/OS UNIX System Services Messages and Codes*, SA22-7807
- *z/OS UNIX System Services Planning*, GA22-7800
- *z/OS UNIX System Services Programming: Assembler Callable Services Reference*, SA22-7803
- *z/OS UNIX System Services User's Guide*, SA22-7801

Index

Special characters

, (comma) in DB2 commands 10
: (colon) in DB2 commands 11
′ (apostrophe) in DB2 commands 10
″ (quotation mark) in DB2 commands 10
( ) (parentheses) in DB2 commands 10
* (asterisk)
  DISPLAY THREAD command 193
  FREE PACKAGE command 269
  in DB2 commands 11
  REBIND PACKAGE command 87
. (asterisk)
  DISPLAY PROCEDURE command 184
  START PROCEDURE command 378
  STOP FUNCTION SPECIFIC command 410
  STOP PROCEDURE command 416
= (equal sign) in DB2 commands 10

A

ABEND subcommand of DSN 217
ACCESS option
  START DATABASE command 355
  START DB2 command 362
accounting
  See also ACCTG option
  audit trace, class descriptions 389
  trace
    displaying 205
    starting 383
    stopping 421
ACCTG option
  DISPLAY TRACE command 207
  MODIFY TRACE command 296
  START TRACE command 386
  STOP TRACE command 423
ACQUIRE option
  BIND PLAN subcommand 63
  DSNH command 254
  REBIND PLAN subcommand 63
ACTION option
  BIND PACKAGE subcommand 64
  BIND PLAN subcommand 64
  DCLGEN subcommand 107
  DSNH command 254
  RECOVER INDOUBT command 314
  STOP FUNCTION SPECIFIC command 410
  STOP PROCEDURE command 416
ACTIVE option
  DISPLAY BUFFERPOOL command 122
  DISPLAY DATABASE command 137
  DISPLAY THREAD command 194
ADD option
  DCLGEN subcommand 108
  DSNH command 257
administrative scheduler
  commands
    MODIFY admtproc,APPL=SHUTDOWN option 273
    MODIFY admtproc,APPL=TRACE option 275
    START admtproc option 345
administrative scheduler *(continued)*
  commands *(continued)*
    STOP admtproc option 347
ADVISORY option of DISPLAY DATABASE command 138
AFTER option of DISPLAY DATABASE command 137
ALL keyword of MODIFY irlmproc,DIAG command 280
ALLD option of MODIFY irlmproc,STATUS command of z/OS 288
ALLI option of MODIFY irlmproc,STATUS command of z/OS 288
ALTER BUFFERPOOL command
  description 29
  example 34
  option descriptions 30
ALTER GROUPBUFFERPOOL command
  description 35
  example 38
  option descriptions 36
ALTER UTILITY command
  description 39
  example 41
ambiguous cursor 67
APOST option
  DCLGEN subcommand 109
  DSNH command 246
APOSTSQL option of DSNH command 252
application package
  See package
application plan
  binding 57
  deleting 271
  maximum size 5
  rebinding, changing plans 303
application program
  START command 349
  testing 217
application program, preparing for DSNH CLIST processing 235
ARCHIVE LOG command
  description 43
  example 47
  option descriptions 44
ASMLIB option of DSNH command 241
ASMLOAD option of DSNH command 241
asterisk (*.*)
  DISPLAY PROCEDURE command 184
  START PROCEDURE command 378
  STOP FUNCTION SPECIFIC command 410
  STOP PROCEDURE command 416
asterisk (*)
  DISPLAY THREAD command 193
  FREE PACKAGE command 269
  in DB2 commands 11
  REBIND PACKAGE command 87
AT option of DCLGEN subcommand 107
AT(COMMIT) option of STOP DATABASE command 400
AUDIT option
  DISPLAY TRACE command 207
  MODIFY TRACE command 296
  START TRACE command 386
  STOP TRACE command 423

© Copyright IBM Corp. 1983, 2008

audit trace
class descriptions 389
displaying 205
starting 383
stopping 421
AUTHID option
DISPLAY TRACE command 208
START TRACE command 388
STOP TRACE command 424
authorization ID
naming convention viii
secondary privileges 3
SQL, privileges exercised by 3
AUTOREC option of ALTER GROUPBUFFERPOOL command 36

B

BDBRMLIB option of DSNH command 255
BDMEM option of DSNH command 254
BIND PACKAGE subcommand of DSN
See also DSN command of TSO
description 49
example 54
option descriptions 63
BIND PLAN subcommand of DSN
See also DSN command of TSO
description 57
example 60
option descriptions 63
binding
See also BIND PACKAGE
See also BIND PLAN
See also REBIND PACKAGE
See also REBIND PLAN
DSNH processing 235
initiating 49, 57
options for 63
blank characters in DB2 command 10
BLIB option of DSNH command 254
BMEM option of DSNH command 255
BnLIB option of DSNH command 254
bootstrap data set (BSDS)
See BSDS (bootstrap data set)
BSDS (bootstrap data set), recovery 311
buffer pool
active and inactive 29, 121
altering attributes 29
displaying current status 121
parallel sequential steal threshold (VPSEQT) 31
BUFSIZE option of START TRACE command 391

C

C option of DCLGEN subcommand 108
CACHESIZE option
BIND PLAN subcommand 65
DSNH command 255
REBIND PLAN subcommand 65
CANCEL OFFLOAD option of ARCHIVE LOG command 46
CANCEL option of RECOVER POSTPONED command 318
CANCEL THREAD command
description 97
example 101
option descriptions 97
cancelling threads, description 97
CASTOUT option of STOP DB2 command 404
CATENFM utility, effects of TERM command 428
CATMAINT utility, effects of TERM command 429
CCLINK option of DSNH command 241
CCLLIB option of DSNH command 241
CCLOAD option of DSNH command 241
CCMSGS option of DSNH command 241
CCOLIB option of DSNH command 241
CCPLIB option of DSNH command 241
CCPMSGS option of DSNH command 241
CCSID option of DSNH command 241
CCSLIB option of DSNH command 241
CHANGE command of IMS
description 103
example 104
character vii
CHECK DATA utility, effects of TERM command 429
CHECK INDEX utility, effects of TERM command 429
CHECK LOB utility, effects of TERM command 429
CHKTIME option of SET LOG command 336
CICS
commands
DSNC 221
DSNC DISCONNECT 223
DSNC DISPLAY 225
DSNC MODIFY 229
DSNC STOP 231
DSNC STRT 233
option of BIND and REBIND subcommands 75
option of DSNH command 255
translation step in DSNH processing 235
CICSCOB option of DSNH command 242
CICSLLIB option of DSNH command 242
CICSOPT option of DSNH command 241
CICSPLIB option of DSNH command 242
CICSPRE option of DSNH command 242
CICSVER option of DSNH command 242
CICSXLAT option of DSNH command 242
CLAIMERS option of DISPLAY DATABASE command 135
CLASS option
DISPLAY TRACE command 209
IFCIDs activated by trace class 388
MODIFY TRACE command 296
START TRACE command 388
STOP TRACE command 424
CLASST option of ALTER GROUPBUFFERPOOL command 37
CLIB option of DSNH command 242
CnLIB option of DSNH command 242
COB2 option of DCLGEN subcommand 108
COB2CICS option of DSNH command 243
COB2LIB option of DSNH command 243
COB2LOAD option of DSNH command 243
COBICOMP option of DSNH command 242
COBILINK option of DSNH command 242
COBIPLNK option of DSNH command 242
COBIPMSG option of DSNH command 242
COBLIB option of DSNH command 243
COBLOAD option of DSNH command 243
COBOL option of DCLGEN subcommand 108
COBSOM option of DSNH command 243
code, return
See return code
collection, package
BIND PACKAGE subcommand 86
ID naming convention viii
REBIND PACKAGE subcommand 86

collection, package *(continued)*
REBIND TRIGGER PACKAGE subcommand 308
COLSUFFIX option of DCLGEN subcommand 109
column name, as a field name 109
comma option of DSNH command 245
command continuation character 19
command prefix
description 9
multiple subsystems 363
part of a command 9
command recognition character (CRC)
See CRC (command recognition character)
commands, scope 13
comment
DCLGEN subcommand output 110
DSN subcommands 217
COMMENT option
DISPLAY TRACE command 208
MODIFY TRACE command 297
START TRACE command 386
STOP TRACE command 424
commit point, terminating utility 427
COMP option of TRACE CT command 434
COMPILE option of DSNH command 243
conditional restart
control record, effect on restart 363
CONNECT option of DSN command 243
connection
DB2
GROUP option of DSN command 218
RETRY option of DSN command 218
displaying
connection information 191
group buffer pool 171
IRLM subsystem status 289
status 115
DSNC DISPLAY command 226
terminating 395
connection-name naming convention viii
CONNLIST option of DISPLAY GROUPBUFFERPOOL command 165
CONTROL option of DSNH command 243
COPTION option of DSNH command 243
COPY option
BIND PACKAGE subcommand 66
DSNH command 257
COPY utility, effects of TERM command 429
COPYVER option
BIND PACKAGE subcommand 66
DSNH command 257
correlation ID
naming convention viii
recovering threads 314
CORRELATION option of START TRACE command 392
COUNT option of SET ARCHIVE command 332
CP option of RUN subcommand 328
CPP option of DCLGEN subcommand 108
CPPCLASS option of DSNH command 243
CPPCLLIB option of DSNH command 244
CPPCSLIB option of DSNH command 244
CPPLINK option of DSNH command 244
CPPLLIB option of DSNH command 244
CPPPMSGS option of DSNH command 244
CPPSLIB option of DSNH command 244
CPPUTIL option of DSNH command 244
CPU option of START TRACE command 392
CRC (command recognition character), description 9
CURRENTDATA option
BIND PACKAGE subcommand 67
BIND PLAN subcommand 67
DSNH command 255, 257
REBIND PACKAGE subcommand 67
REBIND PLAN subcommand 67
REBIND TRIGGER PACKAGE subcommand 67
CURRENTSERVER option
BIND PLAN subcommand 68
DSNH command 255
REBIND PLAN subcommand 68
cursor, ambiguous 67
CYLINDER option of DSNH command 252

D

data set, naming convention viii
data sharing
delays, diagnosing 279
displaying
archive log information 119
information about groups 157
status of members 157
identifying members with utility jobs 212
scope of commands 13
starting members 364
database
displaying status 131
reserved names 400
starting 351
stopping 397
database request module (DBRM)
See DBRM (database request module)
DATE
option of DSNH command 244
DB2 commands
command names 10
commands
ALTER BUFFERPOOL 29
ALTER GROUPBUFFERPOOL 35
ALTER UTILITY 39
ARCHIVE LOG 43
CANCEL THREAD 97
DISPLAY ARCHIVE 119
DISPLAY BUFFERPOOL 121
DISPLAY DATABASE 131
DISPLAY DDF 149
DISPLAY GROUP 157
DISPLAY GROUPBUFFERPOOL 163
DISPLAY LOCATION 177
DISPLAY LOG 181
DISPLAY PROCEDURE 183
DISPLAY RLIMIT 189
DISPLAY THREAD 191
DISPLAY TRACE 205
DISPLAY UTILITY 211
MODIFY TRACE 295
RECOVER BSDS 311
RECOVER INDOUBT 313, 316
RECOVER POSTPONED 317
RESET INDOUBT 323
SET SYSPARM 339
START DATABASE 351
START DB2 361
START DDF 365
START FUNCTION SPECIFIC 367
START PROCEDURE 377

DB2 commands *(continued)*
START RLIMIT 381
START TRACE 383
STOP DATABASE 397
STOP DB2 403
STOP DDF 405
STOP FUNCTION SPECIFIC 409
STOP PROCEDURE 415
STOP RLIMIT 419
STOP TRACE 421
TERM UTILITY 427
completion messages 23
description of 10
DISPLAY FUNCTION SPECIFIC 153
entering from supported environments 21
scope 13
separator 10
DB2 precompiler 10
DBM1 option of START DB2 command 363
DBPROTOCOL option
BIND PACKAGE subcommand 68
BIND PLAN subcommand 68
DSNH command 255
REBIND PACKAGE subcommand 68
REBIND PLAN subcommand 68
DBRM (database request module)
BIND PLAN subcommand 84
maximum number in plan 5
DBRMLIB option of DSNH command 245
DCLGEN (declarations generator)
See DCLGEN subcommand of DSN
DCLGEN subcommand of DSN
See also DSN command of TSO
declaring an indicator variable array 110
description 105
example 112
forming field names 109
option descriptions 106
DDF (distributed data facility), displaying 149
DEADLINE option of ALTER UTILITY command 40
DEADLOK option of START irlmproc command 372
DECARTH option of DSNH command 245
DECIMAL
option of DSNH command 245
declarations
See DCLGEN subcommand of DSN
DEFAULT option of SET ARCHIVE command 332
DEFER
option of BIND PLAN subcommand 69
DEFER option
BIND PACKAGE subcommand 69
BIND PLAN subcommand 69
DSNH command 255
REBIND PACKAGE subcommand 69
REBIND PLAN subcommand 69
degree of parallel processing 70
DEGREE option
BIND PACKAGE subcommand 70
BIND PLAN subcommand 70
DSNH command 255
REBIND PACKAGE subcommand 70
REBIND PLAN subcommand 70
DELAY
keyword of MODIFY irlmproc,DIAG command 280
DELAY option of ALTER UTILITY command 41
deleting, IMS units of recovery 103
DELIMIT option of DSNH command 246
deregistering IRLM 278
DEST option
DISPLAY TRACE command 208
START TRACE command 386
STOP TRACE command 424
DESTINATION option of DSNC MODIFY command 229
DETAIL option
DISPLAY BUFFERPOOL command 122
DISPLAY GROUP command 158
DISPLAY LOCATION command 178
DISPLAY THREAD command 196
DISPLAY TRACE command 207
detail report of DISPLAY BUFFERPOOL command 125
DIAG keyword of MODIFY irlmproc,DIAG command 279
DIAGNOSE utility, TERM command effects 429
diagnostic dumps, IRLM 279
DISABLE option
BIND PACKAGE subcommand 75
BIND PLAN subcommand 75
DSNH command 255
REBIND PACKAGE subcommand 75
REBIND PLAN subcommand 75
disabling a function permanently 411
DISCONNECT option
BIND PLAN subcommand 71
DSNH command 255
REBIND PLAN subcommand 71
DISPLAY ARCHIVE command 119
DISPLAY BUFFERPOOL command
description 121
option descriptions 122
output 124
DISPLAY command of IMS
description 115
example 117
option descriptions 115
DISPLAY DATABASE command
description 131
example 141
option descriptions 134
DISPLAY DDF command
description 149
example 150
option descriptions 149
output 150
DISPLAY FUNCTION SPECIFIC command
description 153
examples 156
output 155
DISPLAY GROUP command
description 157
examples 160
IRLM information 158
DISPLAY GROUPBUFFERPOOL command
description 163
option descriptions 164
output 166
summary report example 171
DISPLAY LOCATION command
description 177
example 179
option descriptions 178
DISPLAY LOG command
description 181
example 182
DISPLAY NET command of VTAM 99

DISPLAY PROCEDURE command
description 183
example 186
option descriptions 184
output 185
DISPLAY RLIMIT command 189
DISPLAY THREAD command
description 191
example 199
option descriptions 193
output 197
DISPLAY TRACE command
description 205
example 209
option descriptions 207
output 210
DISPLAY UTILITY command
description 211
example 214
option descriptions 211
output 212
displaying
information about
archive logs 119
communications database and resource limit facility 139
data sharing group 157
data-partitioned secondary indexes 140
DB2 functions 153
DB2 threads 191
DDF 149
logical partitions 139
logs 181
resource limit facility (governor) 189
restricted objects 139
stored procedures 183
threads with remote locations 177
trace activity 205
status of
buffer pools 121
DB2 databases 131
DB2 utilities 211
group buffer pools 163
DIST option of START DB2 command 363
DISTRIBUTED option of START TRACE command 392
DLIBATCH option
BIND and REBIND subcommands 76
DSNH command 256
DSN command of TSO
abbreviations 19
description 217
example 219
option descriptions 218
parsing subcommands 19
return codes 219
subcommands
ABEND 217
BIND PACKAGE 49
BIND PLAN 57
DCLGEN 105
END 265
FREE PACKAGE 267
FREE PLAN 271
REBIND PACKAGE 299
REBIND PLAN 303
REBIND TRIGGER PACKAGE 307
RUN 327
DSN command of TSO *(continued)*
subcommands *(continued)*
SPUFI 341
DSNC command of CICS 221
DSNC DISCONNECT command of CICS
description 223
example 224
DSNC DISPLAY command of CICS
description 225
example 228
option descriptions 225
output 226
DSNC MODIFY command of CICS
description 229
example 230
option descriptions 229
DSNC STOP command of CICS
description 231
example 232
option descriptions 231
DSNC STRT command of CICS
description 233
example 234
option descriptions 233
DSNDB01 database, authority needed to start 352
DSNDB06 database, authority needed to start 352
DSNH command of TSO
data set names 240
description 235
example 261
option descriptions 237
DSNHDECP
list of parameters 439
DSNZPARM
list of parameters 439
option of START DB2 command 362
DUMP option
CANCEL THREAD command 98
MODIFY irlmproc,ABEND command 277
dump, IRLM diagnostic 279
DWQT option of ALTER BUFFERPOOL command 31
DYNAMICRULES option
BIND PACKAGE subcommand 71
BIND PLAN subcommand 71
DSNH command 256
REBIND PACKAGE subcommand 71
REBIND PLAN subcommand 71

E

ENABLE option
BIND PACKAGE subcommand 75
BIND PLAN subcommand 75
DSNH command 256
REBIND PACKAGE subcommand 75
REBIND PLAN subcommand 75
ENCODING option
BIND PACKAGE subcommand 77
BIND PLAN subcommand 77
REBIND PACKAGE subcommand 77
REBIND PLAN subcommand 77
END subcommand of DSN
description 265
example 265
ENTRY option of DSNH command 246
escape character
APOST option of DCLGEN subcommand 109

escape character *(continued)*
QUOTE option of DCLGEN subcommand 109
establishing connections between IMS and a subsystem 349
EXPLAIN option
BIND PACKAGE subcommand 78
BIND PLAN subcommand 78
DSNH command 256, 259
REBIND PACKAGE subcommand 78
REBIND PLAN subcommand 78
REBIND TRIGGER PACKAGE subcommand 78
extended MCS consoles, DB2 support of 22

F

FLAG option
BIND PACKAGE subcommand 79
BIND PLAN subcommand 79
DSNH command 246, 256
FREE PACKAGE subcommand 269
FREE PLAN subcommand 272
REBIND PACKAGE subcommand 79
REBIND PLAN subcommand 79
REBIND TRIGGER PACKAGE subcommand 79
FORCE option
DSNC STOP command 231
RESET INDOUBT command 324
START DATABASE command 355
STOP DB2 command 404
STOP DDF command 406
FORTLIB option of DSNH command 246
FORTLOAD option of DSNH command 246
FREE PACKAGE subcommand of DSN
See also DSN command of TSO
description 267
example 269
option descriptions 268
FREE PLAN subcommand of DSN
See also DSN command of TSO
description 271
example 272
option descriptions 272
functions, displaying information about 153

G

GBPCACHE option of ALTER GROUPBUFFERPOOL command 36
GBPCHKPT option of ALTER GROUPBUFFERPOOL command 37
GBPOOLT option of ALTER GROUPBUFFERPOOL command 37
GDETAIL option of DISPLAY GROUPBUFFERPOOL command 165
GLOBAL option of START irlmproc command 374
glossary 451
GRAPHIC option of DSNH command 246
group buffer pool RECOVER-pending (GRECP) status,removing using START DATABASE command 356
group detail report of DISPLAY GROUPBUFFERPOOL command 166
GROUP option of DSN command 218
group, scope of command 13
GTF option
DISPLAY TRACE command 208
START TRACE command 387
STOP TRACE command 424

H

HOST option of DSNH command 247

I

I/O processing, parallel, DEGREE option of bind subcommands 70
IBMCOB option of DCLGEN subcommand 108
ID option
RECOVER INDOUBT command 314
START RLIMIT command 381
IFCID (instrumentation facility component identifier), identifiers by trace class 388
IFCID option
MODIFY TRACE command 297
START TRACE command 391
IMMEDWRITE option
BIND PACKAGE subcommand 80
BIND PLAN subcommand 80
REBIND PACKAGE subcommand 80
REBIND PLAN subcommand 80
IMS
commands
CHANGE 103
DISPLAY 115
SSR 343
START 349
STOP 395
TRACE 431
facilities, events tracing 431
IMSBMP option
BIND and REBIND subcommands 76
DSNH command 256
IMSMPP option
BIND and REBIND subcommands 76
DSNH command 256
IMSPRE option of DSNH command 247
INACTIVE option of DISPLAY THREAD command 194
INCLUDE statement of DCLGEN subcommand output 110
indicator variable, array declaration in DCLGEN 110
INDOUBT option of DISPLAY THREAD command 194
indoubt thread, recovering 313
INDVAR option of DCLGEN subcommand 110
INPUT option of DSNH command 247
INTERVAL option of DISPLAY BUFFERPOOL command 123
invalidated plans and packages 79
IPADDR option
DISPLAY LOCATION command 178
RESET INDOUBT command 325
IRLM (internal resource lock manager)
commands
MODIFY irlmproc,ABEND 277
MODIFY irlmproc,DIAG 279
MODIFY irlmproc,PURGE option 281
MODIFY irlmproc,SET option 283
MODIFY irlmproc,STATUS option 287
START irlmproc 371
STOP irlmproc 413
TRACE CT 433
CSA
setting maximum amount of 283
delays, diagnosing 279
deregistering 277
diagnostic dumps 279
DISPLAY GROUP command output 158
locks, releasing 281

IRLM (internal resource lock manager) *(continued)*
modifying, diagnostic trace 433
overview 25
restarting
effect on CSA value 285
starting
an IRLM component 371
diagnostic trace 433
status
checking 283
status, checking 287
stopping
diagnostic trace 433
normal 413
terminating
abnormal 277
normal 413
trace buffers, setting number of 283
IRLMGRP option of START irlmproc command 372
IRLMID option of START irlmproc command 372
IRLMNM option of START irlmproc command 373
ISOLATION
option of BIND PLAN subcommand
description 81
ISOLATION option
BIND PACKAGE subcommand 81
DSNH command 256
REBIND PACKAGE subcommand 81
REBIND PLAN subcommand 81
REBIND TRIGGER PACKAGE subcommand 81

K

KEEPDYNAMIC option
BIND PACKAGE subcommand 83
BIND PLAN subcommand 83
DSNH command 256
REBIND PACKAGE subcommand 83
REBIND PLAN subcommand 83

L

LABEL option of DCLGEN subcommand 109
LANGUAGE option of DCLGEN subcommand 108
LEAVE option of DSNH command 253
letter, description in DB2 vii
LIBRARY option
BIND PACKAGE subcommand 84
BIND PLAN subcommand 84
DCLGEN subcommand 107
RUN subcommand 328
LIGHT option, START DB2 command 362
light restart, with ARM 363
LIMIT option of DISPLAY DATABASE command 137
LINECOUNT option of DSNH command 247
LINK option of DSNH command 247
link-editing, processing 235
LIST option of DISPLAY BUFFERPOOL command 123
LLIB option of DSNH command 247
LnLIB option of DSNH command 247
LOAD option
DSNH command 247
SET SYSPARM command 339
LOAD utility, effects of TERM command 429
LOCAL option of START irlmproc command 374
location name
BIND PACKAGE subcommand 76, 86
DISPLAY LOCATION command 178
REBIND PACKAGE subcommand 76, 86
REBIND TRIGGER PACKAGE subcommand 308
LOCATION option
DISPLAY THREAD command 194
DISPLAY TRACE command 209
RESET INDOUBT command 324
START TRACE command 392
STOP TRACE command 425
LOCKS option of DISPLAY DATABASE command 136
LOCKTABL option of START irlmproc command 373
LOG option of TRACE command 432
logical page list (LPL)
See LPL (logical page list)
logical partitions, displaying 139
LOGLOAD option of SET LOG command 336
LONGLOG option of ALTER UTILITY option 41
LOPTION option of DSNH command 247
LPL (logical page list)
option of DISPLAY DATABASE command 136
recovering pages
using START DATABASE command 356
LSTATS option of DISPLAY BUFFERPOOL command 123
LTE option of MODIFY irlmproc,SET command of z/OS 284
LTE option of START irlmproc command 373
LUNAME option
DISPLAY LOCATION command 178
RESET INDOUBT command 324
LUWID option
DISPLAY THREAD command 195
RECOVER INDOUBT command 315
RESET INDOUBT command 325

M

MACRO option of DSNH command 248
MAINT option of MODIFY irlmproc,STATUS command of z/OS 288
MAINT option of START DB2 command 362
MAXCSA option of START irlmproc command 373
MAXRO option of ALTER UTILITY command 40
MAXUSRS option of START irlmproc command 373
MCS consoles, scope of commands 13
MDETAIL option of DISPLAY GROUPBUFFERPOOL command 165
member detail report of DISPLAY GROUPBUFFERPOOL command 167
MEMBER option
BIND PACKAGE subcommand 84
BIND PLAN subcommand 84
DISPLAY UTILITY command 212
member, scope of command 13
MERGECOPY utility, effects of TERM command 429
message
DB2 commands 23
DCLGEN subcommand 108
DISPLAY THREAD with ACTIVE 194
DISPLAY TRACE command 210
DISPLAY UTILITY command 212
DSN command of TSO 218
DSNH command 246
FLAG option of bind subcommands 79
FREE PACKAGE subcommand 269
FREE PLAN subcommand 272
MODIFY irlmproc,STATUS command 289

message *(continued)*
RUN subcommand 329
message by identifier
DSN7106I 158
DSN9022I 23
DSN9023I 23
DSNB411I 125
DSNB412I 125
DSNB413I 126
DSNB414I 126
DSNB415I 126
DSNB420I 126
DSNB421I 126
DSNI021I 358
DSNJ315I 46
DSNJ316I 46
DSNJ317I 46
DSNJ318I 46
DSNL440I to DSNL449I 325
DSNL448I 324
DSNL450I 98
DSNT392I 140
DSNT500I 141
DSNT501I 141
DSNT736I 401
DSNU100I 212
DSNU105I 212
DSNU106I 212
DSNV413I 197
DSNW133I 386
DSNX943I 185
DSNX950I 185
MLT option of MODIFY irlmproc,SET command of z/OS 284
MODE option
ARCHIVE LOG command 44
STOP DB2 command 403
STOP DDF command 406
MODIFY admtproc
shutdown 273
trace 275
MODIFY admtproc,APPL=SHUTDOWN command of z/OS
description 273
example 273
MODIFY admtproc,APPL=TRACE command of z/OS
description 275
example 276
MODIFY irlmproc,ABEND command of z/OS
description 277
example 278
MODIFY irlmproc,DIAG command of z/OS
description 279
example 280
MODIFY irlmproc,PURGE command of z/OS
description 281
example 282
MODIFY irlmproc,SET command of z/OS
description 283
example 286
option descriptions 281, 283
MODIFY irlmproc,STATUS command of z/OS
description 287
example 289
option descriptions 287
MODIFY RECOVERY utility, effects of TERM command 429
MODIFY STATISTICS utility, effects of TERM command 429
MODIFY TRACE command
description 295
MODIFY TRACE command *(continued)*
example 297
MONITOR option
DISPLAY TRACE command 207
MODIFY TRACE command 296
START TRACE command 386
STOP TRACE command 423
monitor trace
class descriptions 390
displaying 205
starting 383
stopping 421
MSTR option of START DB2 command 363

N

NAMES option of DCLGEN subcommand 108
naming convention, variables in command syntax vii
NEWFUN option of DSNH command 248
NID (network ID) option of RECOVER INDOUBT command 315
NO LIMIT option of SET ARCHIVE command 332
NO option
START DB2 command 362
START irlmproc command 375
NOBACKOUT option of CANCEL THREAD command 98
NODEFER option
BIND PACKAGE subcommand 69
BIND PLAN subcommand 69
DSNH command 256
REBIND PACKAGE subcommand 69
REBIND PLAN subcommand 69
NODISCON option of START irlmproc command 374
NODUMP option of MODIFY irlmproc,ABEND command 277
NOFOR option of DSNH command 248
NONE keyword of MODIFY irlmproc,DIAG command 280
notices, legal 447
NOWRAP option of TRACE CT command 434

O

OASN option
CHANGE command 104
DISPLAY command 116
ONLY option of DISPLAY DATABASE command 136
OP option
START TRACE command 387
STOP TRACE command 424
operands, DB2 commands 10
OPTHINT option
BIND PACKAGE subcommand 85
BIND PLAN subcommand 85
DSNH command 256
REBIND PACKAGE subcommand 85
REBIND PLAN subcommand 85
OPTIONS option
BIND PACKAGE subcommand 85
DSNH command 248
OUTNAME option of DSNH command 248
OVERVIEW option of DISPLAY DATABASE command 137
OWNER option
BIND PACKAGE subcommand 86
BIND PLAN subcommand 86
DCLGEN subcommand 107
DSNH command 256

OWNER option *(continued)*
REBIND PACKAGE subcommand 86
REBIND PLAN subcommand 86

P

P irlmproc command.
See STOP irlmproc command of z/OS
package
binding, initiating 49
identifier
BIND PACKAGE subcommand 86
REBIND TRIGGER PACKAGE subcommand 308
rebinding 299
rebinding trigger 307
replacing version of 65
PACKAGE option of DSNH command 259
PACTION option of DSNH command 257
parallel processing
DEGREE option of bind subcommands 70
VPPSEQT option of ALTER BUFFERPOOL command 31
parameter, passing to application program 328
PARM option of START DB2 command 362
PARMS option
DSNH command 248
RUN subcommand 328
parsing rules, DB2 commands 9
PART option
DISPLAY DATABASE command 136
START DATABASE command 355
STOP DATABASE command 400
partial-location name, DISPLAY LOCATION command 178
PASS option of DSNH command 248
PATH option
BIND PACKAGE subcommand 88
BIND PLAN subcommand 88
DSNH command 256
REBIND PACKAGE subcommand 88
REBIND PLAN subcommand 88
PATHDEFAULT option
REBIND PACKAGE subcommand 89
REBIND PLAN subcommand 89
PBIND option of DSNH command 257
PC option of START irlmproc command 374
PCICS option of DSNH command 257
PCLOAD option of DSNH command 248
PDBPROTOCOL option of DSNH command 258
PDBRMLIB option of DSNH command 258
PDEFER option of DSNH command 258
PDEGREE option of DSNH command 258
PDISABLE option of DSNH command 258
PDLIBATCH option of DSNH command 258
PDMEM option of DSNH command 258
PDYNAMICRULES option of DSNH command 258
PENABLE option of DSNH command 259
PERFM option
DISPLAY TRACE command 207
MODIFY TRACE command 296
START TRACE command 385
STOP TRACE command 423
performance trace
displaying 205
stopping 421
performance, trace
class descriptions 390
starting 383
PFLAG option of DSNH command 259
PGPROT option of START irlmproc command 374
PGSTEAL option of ALTER BUFFERPOOL command 32
phases of execution for DSNH processing 235
PIMSBMP option of DSNH command 259
PIMSMPP option of DSNH command 259
PISOLATION option of DSNH command 259
PKEEPDYNAMIC option of DSNH command 259
PKLIST option
BIND PLAN subcommand 89
DSNH command 256
REBIND PLAN subcommand 89
PL/I application program, macro processing step for DSNH 235
plan
See application plan
PLAN
option of DSNH command 249
PLAN option
BIND PLAN subcommand 90
DISPLAY TRACE command 208
DSNC DISPLAY command 225
DSNH command 257
REBIND PLAN subcommand 90
RUN subcommand 328
START TRACE command 388
STOP TRACE command 424
plan-name naming convention x
PLI option of DCLGEN subcommand 108
PLI2LIB option of DSNH command 249
PLIB option of DSNH command 249
PLILIB option of DSNH command 249
PLILOAD option of DSNH command 249
PLIPLNK option of DSNH command 249
PLIPMSG option of DSNH command 249
PLOCK keyword of MODIFY irlmproc,DIAG command 280
PnLIB option of DSNH command 249
PNODEFER option of DSNH command 259
POPTHINT option of DSNH command 259
POPTION option of DSNH command 249
POSTPONED option of DISPLAY THREAD command 194
postponed units of recovery, recovering 317
POWNER option of DSNH command 259
PPATH option of DSNH command 259
PQUALIFIER option of DSNH command 259
PRECOMP option of DSNH command 249
precompiler
DSNH command options 248
invoking DSNH 235
producing members for 84
PRELEASE option of DSNH command 260
PRELINK option of DSNH command 250
PREOPT option of DSNH command 260
PRINT option of DSNH command 250
privilege set of a process 3
procedure, stored
See stored procedure
processing, parallel
VPPSEQT option of ALTER BUFFERPOOL command 31
PROGRAM option of RUN subcommand 327
PSECSPAC option of DSNH command 250
PSPACE option of DSNH command 250
PVALIDATE option of DSNH command 260
PVT option of MODIFY irlmproc,SET command of z/OS 284

Q

QUALIFIER option
  BIND PACKAGE subcommand 91
  BIND PLAN subcommand 91
  DSNH command 257
  REBIND PACKAGE subcommand 91
  REBIND PLAN subcommand 91
qualifier-name naming convention x
QUEUE option of STOP FUNCTION SPECIFIC 410, 411
QUIESCE option
  DSNC STOP command 231
  STOP DB2 command 404
  STOP DDF command 406
QUIESCE utility, effects of TERM command 429
QUOTE option
  DCLGEN subcommand 109
  DSNH command 246

R

RATIO option of ALTER GROUPBUFFERPOOL command 36
RCTERM option of DSNH command 250
REBIND PACKAGE subcommand of DSN
  See also DSN command of TSO
  description 299
  example 302
  option descriptions 63
REBIND PLAN subcommand of DSN
  See also DSN command of TSO
  description 303
  example 306
  option descriptions 63
REBIND TRIGGER PACKAGE subcommand of DSN
  description 307
  example 309
  option descriptions 63
rebinding
  initiating 299, 303
  options for 63
REBUILD INDEX utility, TERM command effects 429
recognition character 9
RECOVER BSDS command
  description 311
  example 312
RECOVER INDOUBT command
  description 313
  example 316
  option descriptions 314
RECOVER POSTPONED command
  description 317
  example 318
RECOVER utility, TERM command effects 429
recovery
  BSDS 311
  indoubt threads 313
  postponed units of recovery 317
REJECT option of STOP FUNCTION SPECIFIC 410, 411
RELEASE option
  BIND PACKAGE subcommand 91
  BIND PLAN subcommand
    description 91
  DSNH command 257
  REBIND PACKAGE subcommand 91
  REBIND PLAN subcommand 91
  REBIND TRIGGER PACKAGE subcommand 91
RELOAD option of SET SYSPARM command 340
REMOTE option of DSNH command 260
REOPT option
  BIND PACKAGE subcommand 92
  BIND PLAN subcommand 92
  DSNH command 257
  REBIND PACKAGE subcommand 92
  REBIND PLAN subcommand 92
REORG INDEX utility, effects of TERM command 429
REORG TABLESPACE utility, effects of TERM command 429
REPAIR utility, effects of TERM command 429
REPLACE option
  DCLGEN subcommand 108
  DSNH command 254, 257
replacing, version of a package 65
REPLVER option
  BIND PACKAGE subcommand 64
  DSNH command 260
  effect of 65
REPORT utility, effects of TERM command 429
reports
  detail report 125
  group detail report 166
  member detail report 167
  summary report
    DISPLAY BUFFERPOOL command 124
    DISPLAY GROUPBUFFERPOOL command 171
RES option of STOP TRACE command 424
RESET GENERICLU command
  description 321
  example 322
  option descriptions 322
RESET INDOUBT command
  description 323
  option descriptions 324
RESET option of CHANGE command 104
resource limit facility
  displaying 139
restarting
  CICS attachment facility 232
  connections between IMS and a subsystem 349
  status of DB2 resources 361
  terminated utility job steps 428
RESTRICT option of DISPLAY DATABASE command 137
RESUME option of SET LOG command 336
RETAIN option
  BIND PLAN subcommand 64
  DSNH command 257
retained locks 136
RETRY option of DSN command 218
return code
  CURRENTSERVER option of bind and rebind
    subcommands 68
  DSN command 219
  RUN subcommand of DSN 219
RO option of START DATABASE command 355
RRSURID option of DISPLAY THREAD command 196
RUN
  subcommand of DSN
    description 327
    example 329
    option descriptions 327
    return codes 219
RUN option
  DSNH command parameters 251
RUNIN option of DSNH command 251
running DSNH processing 235
RUNOUT option of DSNH command 251

RUNSTATS utility, effects of TERM command 429
RW option of START DATABASE command 355

S

scanning rules, DB2 commands 9
schema.partial-name option
  DISPLAY FUNCTION SPECIFIC command 154
  START FUNCTION SPECIFIC command 368
  STOP FUNCTION SPECIFIC command 410
schema.specific-function-name option
  DISPLAY FUNCTION SPECIFIC command 154
  START FUNCTION SPECIFIC command 368
  STOP FUNCTION SPECIFIC command 410
schema.specific-function-name option of DISPLAY FUNCTION SPECIFIC command 154
scope of commands 13
SCOPE option
  ARCHIVE LOG command 45
  DISPLAY FUNCTION SPECIFIC command 154
  DISPLAY PROCEDURE command 184
  DISPLAY THREAD command 193
  DISPLAY TRACE command 208
  START FUNCTION SPECIFIC command 368
  START irlmproc command 374
  START PROCEDURE command 378
  STOP FUNCTION SPECIFIC command 411
  STOP PROCEDURE command 417
  STOP TRACE command 424
secondary authorization ID 3
SET ARCHIVE command
  description 331
  example 333
  option descriptions 332
SET LOG command
  description 335
  example 337
  option descriptions 336
SET SYSPARM command
  description 339
  example 340
  option descriptions 339
SMF option
  DISPLAY TRACE command 208
  START TRACE command 387
  STOP TRACE command 424
SOMDLLI option of DSNH command 251
SOURCE option of DSNH command 251
SPACENAM option
  DISPLAY DATABASE command 134
  START DATABASE command 354
  STOP DATABASE command 399
SPACEUN option of DSNH command 252
special character vii
SPUFI subcommand of DSN
  See DSN command of TSO
SPUFI, description 341
SQL ID (SQL authorization ID)
  See authorization ID, SQL
SQL option of DSNH command 252
SQLDELIM option of DSNH command 252
SQLERROR option
  BIND PACKAGE subcommand 93
  DSNH command 260
SQLFLAG option of DSNH command 252
SQLRULES option
  BIND PLAN subcommand 94
SQLRULES option *(continued)*
  REBIND PLAN subcommand 94
SQLRULES option of DSNH command 257
SRC (subsystem recognition character)
  See command prefix
SRV option
  DISPLAY TRACE command 208
  START TRACE command 387
  STOP TRACE command 424
SSR command of IMS, description 343
START admtproc
  trace 345
START admtproc command of z/OS
  description 345
  example 346
START command of IMS 349
START DATABASE command
  description 351
  example 358
  option descriptions 353
  recovering object in group buffer pool 356
  recovering pages on logical page list 356
START DB2 command
  description 361
  example 364
  option descriptions 362
START DDF command 365
START FUNCTION SPECIFIC command
  description 367
  example 369
  option descriptions 368
START irlmproc command of z/OS
  description 371
  examples 375
  option descriptions 372
START PROCEDURE command
  description 377
  example 379
  option descriptions 378
START RLIMIT command
  description 381
  example 382
START TRACE command
  description 383
  example 393
  option descriptions 385
STARTUP option of SET SYSPARM command 340
STAT option
  DISPLAY TRACE command 207
  MODIFY TRACE command 296
  START TRACE command 386
  STOP TRACE command 423
STATISTICS option of DSNC DISPLAY command 226
statistics trace
  stopping 421
statistics, trace
  class descriptions 389
  displaying 205
  starting 383
status
  checking, IRLM 287
  cross-system coupling facility (XCF), status of members 158
  shown by DISPLAY DATABASE 140
STDSQL option of DSNH command 252
STOP admtproc
  trace 347

STOP admtproc command of z/OS
description 347
example 347
STOP command of IMS 395
STOP DATABASE command
description 397
example 402
option descriptions 398
STOP DB2 command
description 403
example 404
STOP DDF command
description 405
example 407
STOP FUNCTION SPECIFIC command
description 409
examples 411
limitations of 411
option descriptions 410
STOP irlmproc command of z/OS 413
STOP PROCEDURE command
description 415
example 417
option descriptions 416
STOP RLIMIT command 419
STOP TRACE command
description 421
example 426
option descriptions 423
STOR option of MODIFY irlmproc,STATUS command of z/OS 288
stored procedure
displaying status 183
starting 377
stopping 415
STOSPACE utility, effects of TERM command 429
string option naming convention x
string, delimiter
COBOL 109
SQL 109
STRUCTURE option of DCLGEN subcommand 108
SUB option of TRACE CT command 434
SUBS option of TRACE command 431
SUBSYS option
CHANGE command 103
DISPLAY command 115
START command 349
STOP command 395
subsystem parameters
list 439
NPGTHRSH 441
PTASKROL 442
subsystem, naming convention x
SUFFIX option of DSNH command 252
summary report
DISPLAY BUFFERPOOL command 124
DISPLAY GROUPBUFFERPOOL command 165, 171
SUSPEND option
SET LOG command 336
STOP DDF command 406
syntax diagram
how to read xi
SYSTEM option
DSN command 218
DSNH command 253

T

table name, naming convention x
TABLE option of DCLGEN subcommand 106
table space, naming convention x
TDATA option of START TRACE command 391
TERM option of DSNH command 253
TERM UTILITY command
description 427
example 429
terminating
connections between IMS and a subsystem 395
databases 397
DB2, description 403
IRLM
abnormal 277
normal 413
stored procedures 415
trace activity 421
utilities, description 427
TEST option of DSN command 218
thread
canceling 97
displaying 191
TIME option
ARCHIVE LOG command 45
DSNH command 253
SET ARCHIVE command 332
TIMEOUT option of MODIFY irlmproc,SET command of z/OS 284
TNO option
DISPLAY TRACE command 209
MODIFY TRACE command 296
trace
changing active traces 295
displaying 205
events 431
IFCIDs activated by trace class 388
starting 383
stopping 421
TRACE command of IMS
description 431
example 432
TRACE CT command of z/OS
description 433
example 435
option descriptions 434
TRACE option
MODIFY irlmproc,SET command of z/OS 284
MODIFY irlmproc,STATUS command of z/OS 289
START irlmproc command 374
START TRACE command 392
TRACK option of DSNH 252
TRANSACTION option
DSNC DISPLAY command 226
DSNC MODIFY command 230
TSO CLISTs of DSNH 235
TSO option of DSNH command 251
TYPE option
DISPLAY GROUPBUFFERPOOL command 164
DISPLAY THREAD command 194

U

unit of recovery, in IMS 116
unit of work
See also unit of recovery

unit of work *(continued)*
displaying an outstanding 115
resetting
IMS 103
indoubt 103
UNLOAD utility, effects of TERM command 429
USE option of DISPLAY DATABASE command 135
UT option of START DATABASE command 355
utilities
displaying status 211
identifier 428
terminating 427
utility-id naming convention x

V

VALIDATE option
BIND PACKAGE subcommand 94
BIND PLAN subcommand 94
DSNH command 257
REBIND PACKAGE subcommand 94
REBIND PLAN subcommand 94
VDWQT option of ALTER BUFFERPOOL command 31
version of a package
BIND PACKAGE subcommand 86
REBIND PACKAGE subcommand 86
VERSION option of DSNH command 253
version-id naming convention x
virtual storage access method (VSAM)
See VSAM (virtual storage access method)
VPPSEQT option of ALTER BUFFERPOOL command 31
VPSEQT option of ALTER BUFFERPOOL command 30
VPSIZE option of ALTER BUFFERPOOL command 30
VPXPSEQT option of ALTER BUFFERPOOL command 31
VSAM (virtual storage access method) password, DCLGEN subcommand 107
VTAM (Virtual Telecommunications Access Method), DISPLAY NET command 99

W

WAIT option of ARCHIVE LOG command 45
WEPR option of DISPLAY DATABASE command 136
WORKUNIT option of DSNH command 253
WRAP option of TRACE CT command 434
WSECPAC option of DSNH command 253
WSPACE option of DSNH command 253
WTRSTART option of TRACE CT command 434
WTRSTOP option of TRACE CT command 434

X

XCF (cross-system coupling facility), status of members 158
XLIB option of DSNH command 253
XREF option of DSNH command 253

Y

YES option
START DB2 command 362
START irlmproc command 375

Z

z/OS commands
MODIFY admtproc,APPL=SHUTDOWN 273
MODIFY admtproc,APPL=TRACE 275
MODIFY irlmproc,ABEND 277
MODIFY irlmproc,DIAG 279
MODIFY irlmproc,PURGE 281
MODIFY irlmproc,SET 283
MODIFY irlmproc,STATUS 287
START admtproc 345
START irlmproc 371
STOP admtproc 347
STOP irlmproc 413
TRACE CT 433
ZPARM
See subsystem parameters

Program Number: 5625-DB2

Printed in USA

SC18-7416-05